

מחסני נתונים

עקרונות, ארכיטקטורה, עיצוב ויישום

רז הייפרמן

הוצאת הוד-עמי
ספרי מחשבים

מחסני נתונים

עקרונות, ארכיטקטורה,
עיצוב ויישום

עורך ראשי: **יצחק עמיהוד**

עריכה: **צור ריכטר-לוין**

עריכה ועיצוב: **קרן לנדאו**

עיצוב עטיפה: **שרון רז**

שמות מסחריים

הודעה

לשם שטף הקריאה כתוב ספר זה בלשון זכר בלבד. ספר זה מיועד לגברים ונשים כאחד ואין בכוונתנו להפלות או לפגוע בציבור המשתמשים/ות.

- טלפון: 09-9564716
- פקס: 09-9571582
- דואר אלקטרוני: Info@hod-ami.co.il
- אתר באינטרנט: http://www.hod-ami.co.il

מחסני נתונים

עקרונות, ארכיטקטורה, עיצוב ויישום

רז הייפרמן

הוצאת הוד-עמי

Data Warehouses
Principles, Architectures, Design and Implementation

By **Raz Heiferman**

Editor: **I. Amihud**

הוצאת הוד-עמי
לספרי מחשבים בע"מ
ת.ד. 6108 הרצליה 46160
טלפון: 09-9564716 פקס: 09-9571582
דואר אלקטרוני: info@hod-ami.co.il

הודפס בישראל 1999

HOD-AMI Ltd.
P.O.B. 6108, Herzliya
ISRAEL, 1999

ISBN 965-361-199-2 מסת"ב

הספר מוקדש באהבה

לבני אייל, עם סיום קורס הקצינים ותחילת מסלולו כמפקד בצה"ל

לבתי איילת, עם סיום לימודי התיכון ולקראת התגייסותה לצה"ל

לאשתי ארנונה, עם קבלת בעלה בחזרה לאחר סיום כתיבת הספר

על המחבר

רז הייפרמן, סמנכ"ל ומנהל חטיבת טכנולוגיות מידע בחברת בזק. בעל תואר שני בחקר ביצועים מביה"ס למינהל עסקים באוניברסיטה העברית בירושלים. בתפקידיו הקודמים שרת כמנהל חטיבת טכנולוגיות ופיתוח עסקים בחברת קונתהל, מנהל אגף מחשוב ובקרה בחברת אופטרוטק (שהתמזגה עם חברת אורבוט ונקראת כיום אורבוטק), מנהל פרויקטים בכיר בחברת טכם, מנהל תחום מערכות לוגיסטיות באגף מדעי הניהול ושירותי מידע בבזק (אז עדיין חלק ממשרד התקשורת). מרצה בנושאי בסיסי נתונים, עיצוב מערכות תוכנה, טכנולוגיות שרת/לקוח ומחסני נתונים במוסדות הוראה שונים וביניהם הפקולטה להנדסה באוניברסיטת ת"א, ביה"ס למקצועות המחשב בצה"ל (ממר"ם לשעבר) ואחרים. כתב את הספרים "**בסיסי נתונים טבלאיים ושפת SQL**", "**ארגון קבצים וניהול נתונים**" וערך את הספר "טכנולוגיות שרת/לקוח", כולם **בהוצאת הוד-עמי**.

תוכן עניינים מקוצר

הקדמה 17

פרק 1: רקע ומושגי יסוד........ 25

פרק 2: ארכיטקטורת מחסן הנתונים........ 49

פרק 3: סקירת מרחב הטכנולוגיות לניתוח מידע........ 63

פרק 4: ניתוח מידע רב-מימדי **(Multi Dimensional Analysis)**........ 83

פרק 5: ארכיטקטורות של מוצרים לניתוח רב-מימדי
OLAP Architectures........ 105

פרק 6: עיצוב רב-מימדי של בסיסי נתונים טבלאיים
(Relational Dimensional Database Design)........ 125

פרק 7: כריית נתונים **(Data Mining)**........ 179

פרק 8: הטכנולוגיה המקבילית בשירות מחסן הנתונים........ 205

פרק 9: סקירת התהליכים העיקריים במחסן הנתונים........ 31

פרק 10: מידע על הנתונים **(Metadata)**........ 259

פרק 11: מרכולי נתונים **(Data Marts)**........ 269

פרק 12: מתודולוגיה ליישום מחסן נתונים........ 283

נספח א׳: מילון מונחים........ 313

נספח ב׳: ביבליוגרפיה........ 335

תוכן העניינים

הקדמה 17

מבנה הספר........ 20
קהל היעד של הספר 23

פרק 1 : רקע ומושגי יסוד 25

מבוא........ 25
המגמות האסטרטגיות העיקריות בטכנולוגיית המידע........ 28
המגמות העיקריות בתחום הארכיטקטורה היישומית 29
המגמות העיקריות בתחומי טכנולוגיית המידע........ 31
אפיון סוגי היישומים בארגון........ 33
כללי 33
יישומים עסקיים תפעוליים (Operational Business Applications)........ 34
יישומים תומכי החלטות (Decision Support Applications)........ 34
השוואת מאפייני שני סוגי היישומים........ 35
המסקנה : שתי סביבות שונות, אבל משולבות 39
השלבים בגיבוש תפיסת מחסן הנתונים 41
כללי 41
השלבים בהתפתחות מערכות לאספקת מידע 41
מחוללי דוחות ושאילתות לגישה לנתונים תפעוליים
(Report Writer and Query Generator)........ 41
סביבות ייעודיות לניתוחים סטטיסטיים במחשבים מרכזיים........ 42
מערכות תומכות החלטה (DSS - Decision Support Systems) 43
מערכות מידע למנהלים (EIS - Executive Information Systems)........ 43
מרכזי מידע (Information Centers)........ 44
סביבות תומכות החלטות מבוססות מחשבי אישיים 45
מחסן הנתונים (Data Warehouse)........ 46
סיכום 47

פרק 2: ארכיטקטורת מחסן הנתונים 49

מבוא ... 49
מהו מחסן נתונים? ... 50
ארכיטקטורת מחסן הנתונים (Data Warehouse Architecture) ... 52
ארכיטקטורת מחסן נתונים ארגוני (Enterprise Data Warehouse Architecture) ... 53
ארכיטקטורת מרכול הנתונים (Data Mart Architecture) ... 56
מחסן נתונים רב-שכבתי (Multi Tier Data Warehouse) ... 57
ארכיטקטורת מאגר נתונים תפעולי (ODS - Operational Data Store) ... 59
מחסן נתונים מדומה (Virtual Data Warehouse) ... 61
סיכום ... 62

פרק 3: סקירת מרחב הטכנולוגיות לניתוח מידע 63

מבוא ... 63
מנתונים דרך מידע ועד להחלטות ... 64
נתונים (Data) ... 65
מידע (Information) ... 66
ידע (Knowledge) ... 67
החלטות (Decisions) ... 68
מה זה ניתוח נתונים ... 69
סיווג תהליכי ניתוח הנתונים - אימות מול גילוי ... 71
סיווג הטכנולוגיות התומכות בניתוח נתונים ... 74
מחוללי שאילתות ודוחות (Query and Report Generators) ... 74
כלי ניתוח רב-מימדיים (OLAP - On Line Analytical Processing) ... 75
כלי כריית נתונים (Data Mining) ... 76
סיווג מרכיבי המידע ... 78
סיווג המשימות האנליטיות ... 79
סיווג היישומים האנליטיים לפי הדרג הארגוני ... 80
מפת המרחב האנליטי (Analytical Space) ... 80
סיכום ... 82

פרק 4: ניתוח מידע רב-מימדי (Multi Dimensional Analysis) 83

מבוא ... 83
מושגי היסוד בניתוח רב-מימדי - עובדות ומימדים ... 84
היררכיות בתוך מימדים ... 87
הקוביה הרב-מימדית ... 89
הצגה מימדית של מידע (Dimensional View) ... 94
פעולות יסוד בניתוח מידע רב-מימדי ... 96
מגבלות גודל הקוביה הרב-מימדית ... 102

סיכום 103

פרק 5: ארכיטקטורות של מוצרים לניתוח רב-מימדי - OLAP Architectures 105

מבוא........ 105
חלופות למימוש ארכיטקטורה למוצר OLAP........ 107
מוצרים המבוססים על בסיסי נתונים רב-מימדיים (MOLAP - Multi dimensional OLAP)........ 110
בסיסי נתונים טבלאיים בעיצוב רב-מימדי (Relational Dimensional Database) 111
מוצרים לניתוח רב-מימדי מבוסס בסיס נתונים טבלאי (ROLAP - Relational OLAP) 113
ארכיטקטורת SQL Cube........ 115
ארכיטקטורת Hyper Cube 115
ארכיטקטורת Desktop Hyper Cube........ 116
ארכיטקטורת Flash Cube 117
ארכיטקטורה רב-מימדית מעורבת (Hybrid OLAP) 117
סקירת מוצרי OLAP על פי מטריצת הארכיטקטורות........ 119
מה הארכיטקטורה המומלצת? 120
כללי Codd להגדרת מוצרים רב-מימדיים........ 121
סיכום 123

פרק 6: עיצוב רב-מימדי של בסיסי נתונים טבלאיים (Relational Dimensional Database Design) 125

מבוא........ 125
מדוע נדרשת טכניקת עיצוב מיוחדת........ 126
שיקולי עיצוב עיקריים בסביבת עיבוד תנועות (OLTP) 126
זמן ביצוע תנועה (Transaction Performance) 126
מורכבות אינה בעיה 127
מסלולי גישה ידועים מראש (Predefined Access Path) 127
התמקדות בנתונים שוטפים ולא היסטוריים........ 127
שיקולי עיצוב עיקריים בסביבת מחסן נתונים 128
נוחות המשתמש 128
ביצועי שאילתות (Query Performance) 128
זמן טעינת הנתונים (Data Load Time) 129
נירמול נתונים (Data Normalization) אינו יתרון בסביבת מחסן נתונים 129
בעיית הביצועים 129
בעיית ניהול מידע דינמי מול מידע היסטורי במחסן הנתונים 131
סיכום ההבדלים העיקריים 133
מושגי היסוד - עובדות ומימדים........ 134

עובדות (Facts) 135
מימדים עסקיים (Business Dimensions) 137
ייצוג יישויות על ידי טבלאות (Tables) 138
סכמת כוכב (Star Schema) 139
מבנה 139
שאילתות כוכב (Star Query) 141
יתרונות עיקריים של סכמת הכוכב 141
מבנה המפתחות של הטבלאות (Key Structures) 142
סוגיות מיוחדות בנושא טבלאות המימדים 144
הירככיות של מימדים (Dimensions Hierarchy) 144
מימד הזמן (Time Dimension) 146
מדוע זהו מימד מיוחס 146
ניהול מימד הזמן ישירות בטבלת העובדות 146
ניהול מפורש של מימד הזמן 146
ריבוי הירככיות במימד הזמן 147
מימדים מנוונים (Degenerate Dimensions) 148
טיפול במימדים גדולים (Big Dimensions) 149
מימדים סיבתיים (Casual Dimensions) 149
ריבוי קשרים בין מימד לעובדות (Multiple Column Dimension) 150
סוגיות מיוחדות בנושא טבלת עובדות 151
הגדרת רמת הפירוט (Granularity) 151
טבלת עובדות נטולת עובדות (Factless Fact Table) 152
ריבוי טבלאות עובדות (Multiple Fact Tables) 153
סכמות פתיתי שלג (Snowflake Schema) 155
ניהול שינויים (Change Management) 158
מדוע יש לנהל שינויים 158
אפיון קצב השינויים 158
טכניקות לטיפול בשינויים במודל הנתונים 159
הוספת תכונות חדשות לטבלאות העובדות או המימדים 159
הוספת מימד חדש למודל 160
ניהול שינויים בהירככיות של המימדים 160
בניית רשומת מימד חדשה 161
ניהול מצב עדכני ומצב נוסף 162
ניהול סיכומים (Aggregate Management) 163
הצורך בניהול סיכומים 163
טכניקות לניהול סיכומים 164
ניהול הסיכומים בטבלת העובדות 164
ניהול טבלאות סיכומים נפרדות 167
יתרונות וחסרונות שיטות ניהול הסיכומים 167
קביעת מספר טבלאות הסיכומים שיש לנהל 168

קביעת המפתחות והעמודות של טבלת הסיכומים 173
זיהוי אוטומטי של טבלאות הסיכומים
(Automatic Aggregate Navigation) 174
חישוב נפח בסיס הנתונים 176
חישוב נפח טבלאות מימדים 177
חישוב נפח טבלת העובדות 177

פרק 7: כריית נתונים (Data Mining) 179

מבוא 179
מהי כריית נתונים 179
תחומי המחקר שתרמו להתפתחות כריית הנתונים 182
סטטיסטיקה 183
אינטליגנציה מלאכותית 183
מערכות לומדות (Machine Learning) 183
לימוד והיסק 184
יישומים, טכניקות, אלגוריתמים ומוצרים לכריית נתונים 184
יישומים (Applications) 185
טכניקות (Techniques) 187
הצגה ויזואלית של נתונים (Visualization Techniques) 187
הסקת חוקים ויחסים בין הנתונים (Association Rules) 188
ניתוח סדר הופעה (Sequence Based Analysis) 188
ניתוח אשכולות (Cluster Analysis) 189
סיווג (Classification) 189
עצי החלטה (Decision Trees) 190
רשתות עצביות (Neural Networks) 193
אומדן (Estimation) 196
טכניקות אחרות 196
טבלת סיכום יתרונות וחסרונות הטכניקות השונות 196
אלגוריתמים ומודלים (Algorithms and Models) 197
מוצרים לכריית נתונים (Data Mining Tools) 197
תהליך כריית הנתונים 198
אינטגרציה בין מחסן הנתונים לכריית הנתונים 201
בעיות עיקריות בכריית הנתונים 202
סיכום 203

פרק 8: הטכנולוגיה המקבילית בשירות מחסן הנתונים 205

מבוא 205
ארכיטקטורות מרובות מעבדים לעיבוד מקבילי (Parallel Systems) 206
ארכיטקטורת SMP (Symmetric Multi Processing) 206
ארכיטקטורת אשכול (Cluster Architecture) 207

ארכיטקטורת MPP (Massively Parallel Processing) 209
ארכיטקטורת NUMA (Non Uniform Memory Architecture) 211
ארכיטקטורת Parallel Sysplex של חברת יבמ 211
עיבוד מקבילי של שאילתות (Parallel Query Processing) 212
מהו עיבוד מקבילי של שאילתה 212
סריקה מקבילית של טבלאות (Parallel Table Scan) 213
צירוף מקבילי של טבלאות (Parallel Table Join) 214
מיון מקבילי (Parallel Table Sort) 215
זרימה מקבילית (Query Pipelining) 216
חלוקת טבלה למחיצות (Table Partitioning) 218
מדוע לחלק למחיצות 219
ביצועים 219
כושר גידול (Table Scalability) 219
ניהול טבלאות גדולות 219
גיבוי ושחזור טבלאות גדולות 220
שיטות לחלוקת טבלה למחיצות 220
הקצאה אופקית למחיצות (Horizontal Table Partitioning) 220
הקצאה לפי טווח ערכים של תכונה (Range Partitioning) 221
הקצאה לפי סדר הגעה (Round Robin Partitioning) 223
הקצאה לפי אלגוריתם (Hash Partitioning) 224
חלוקה למחיצות בשיטה ורטיקלית (Vertical Partitioning) 224
חלוקה למחיצות מבוססת חומרה (Data Striping) 225
מהי השיטה המומלצת לחלוקה למחיצות 226
תמיכת בסיס הנתונים בחלוקה למחיצות 226
טעינה מקבילית (Parallel Load) 227
מבחני ביצועים לסביבת מחסן נתונים 228
סיכום 229

פרק 9: סקירת התהליכים העיקריים במחסן הנתונים 231

מבוא 231
סיווג התהליכים במחסן הנתונים 231
סקירת תהליכי זרימה פנימה (In Flow Process) 233
מבט-על של תהליכי הזרימה פנימה 233
גזירת הנתונים (Data Extraction) 235
מדוע יש חשיבות לגזירת הנתונים מהמערכות התפעוליות 236
קריאת נתוני המקור 237
מתי לבצע את גזירת הנתונים 237
סנכרון נתונים הנגזרים מקבצים שונים 238
שיטת הגזירה (Extraction Method) 238
גזירה מלאה (Full Data Propagation) 239

זיהוי שינויים (Changed Data Identification) 239
אז מהי שיטת הגזירה המומלצת 243
האם לשלב תהליכי מיטוב בתוך תהליכי הגזירה 244
האם להעביר גם נתונים שלא נדרשים 244
כיצד לממש את תהליכי הגזירה 244
בדיקת וניקוי הנתונים (Data Cleaning) 245
בדיקת מרחב הערכים (Domain Inconsistency) 245
בדיקות מול טבלאות (Lookup Table Check) 246
בדיקות אמינות נתונים (Data Integrity) 246
בדיקות לגילוי גרסאות שונות של נתונים (Version Check) 246
מיטוב נתונים (Data Enhancement) 246
הפיכת מידע נסתר ועקיף למידע מפורש וגלוי 247
איחוד רשומות (Record De-duplication) 247
תקנון כתובות (Address Normalization) 248
שינוי מבנה הנתונים (Data Transforming) 248
סטנדרטיזציה בייצוג הפיסי 249
שינוי קודים ואיחוד נתונים 249
העברת מונחים תפעוליים למונחים עסקיים 250
ארגון המונחים 250
השלמת ערכים חסרים וברירות מחדל 250
דה-נורמליזציה של מבנה הנתונים (Data Denormalization) 251
בניית מפתחות מלאכותיים (Key Generation) 251
המרת מפתחות עבור טבלת העובדות 252
טעינת הנתונים למחסן הנתונים (Data Loading) 253
תהליכי הזרימה מעלה (Up Flow Process) 254
הכנת טבלאות הסיכום 255
הכנת אינדקסים עבור טבלאות הסיכום 255
סקירת תהליכי הזרימה מטה (Down Flow Process) 255
סקירת תהליכי הזרימה החוצה (Out Flow Process) 256
סקירת תהליכי הזרימה מידע על הנתונים (Meta Data Flow Process) 256
סיכום 257

פרק 10 : מידע על הנתונים (Metadata) 259

לשם מה דרוש מידע על הנתונים 259
מהו "מידע על הנתונים" 260
חשיבות המידע על הנתונים בסביבת מחסן הנתונים 262
קטגוריות של "מידע על הנתונים" 264
מיפוי וגזירת נתונים (Data Mapping and Extraction) 265
מיפוי למונחים עסקיים 266
היסטוריית גזירת הנתונים 266

טבלאות סיכום (Summarization Metadata)....... 266
מידע כמותי (Volume Oriented Metadata)....... 266
מידע אודות נגישות (Access Pattern Metadata)....... 267
בקרת גישות (Access Authorization)....... 267
מידע מערכת (System Metadata)....... 267
מוצרים לניהול המידע על הנתונים....... 267
סיכום....... 268

פרק 11: מרכולי נתונים (Data Marts) 269

מבוא....... 269
מהו מרכול נתונים....... 270
הגורמים לפופולריות הרבה של תפיסת מרכול הנתונים....... 271
הסיכונים והבעיות בתפיסת מרכול הנתונים....... 272
ארכיטקטורת מרכול הנתונים....... 273
מרכול נתונים בארכיטקטורה דו-שכבתית
(Two Tier Data Mart Architecture)....... 273
מרכול נתונים בארכיטקטורה תלת-שכבתית
(Three Tier Data Mart Architecture)....... 274
סוג בסיס הנתונים של מרכול הנתונים....... 276
שיקולים למימוש מרכול הנתונים בארכיטקטורה תלת-שכבתית....... 277
מודלים לפיתוח והקמת מרכולי נתונים....... 278
פיתוח מרכול נתונים במודל מעלה-מטה (Top Down Model)....... 278
פיתוח מרכול נתונים במודל מטה-מעלה (Bottom Up Model)....... 279
פיתוח מרכול נתונים במודל מקבילי (Parallel Model)....... 280
סיכום....... 281

פרק 12: מתודולוגיה ליישום מחסן נתונים 283

מבוא....... 283
עקרונות המתודולוגיה....... 284
מבנה המתודולוגיה....... 285
השלבים במחזור החיים של מחסן הנתונים (Data Warehouse Life Cycle)....... 286
נושאים לטיפול בכל שלב במחזור החיים....... 288
פירוט השלבים במתודולוגיה....... 290
שלב 1: בחינת הצורך העסקי ומוכנות הארגון....... 290
מהות השלב....... 290
זיהוי התהליכים העסקיים העיקריים
(Business Processes Identification)....... 291
זיהוי סדרי עדיפויות ליישום (Implementation Priorities)....... 292
הערכה ראשונית לגודל מחסן הנתונים....... 292
בחינת המוכנות הארגונית (Organizational Readiness)....... 293

הצגת הצורך העסקי והמוכנות הארגונית להנהלה 293
שלב 2: גיבוש ארכיטקטורת העל........ 294
מהות השלב 294
ייזום פרויקט ארכיטקטורת העל (Project Charter and Work Plan) 295
סקירת הנושאים המטופלים בתוכנית האב........ 296
פונקציונליות 296
הנתונים 298
טכנולוגיה........ 301
תמיכה........ 302
תוכנית עבודה ליישום 304
שלב 3: עיצוב ויישום מקטע (Data Warehouse Increment Implementation). 305
מהות השלב 305
בניית מקטע הבסיס (Baseline)........ 305
עיצוב ובניית מקטע 306
פונקציונליות 306
נתונים 306
ארכיטקטורה טכנולוגית........ 307
תמיכה........ 307
שלב 4: תפעול, תמיכה ושיפור 307
מהות השלב 307
מספר עצות מעשיות בבניית מחסן הנתונים 307
השיקולים להקמת מחסן הנתונים חייבים להיגזר מאסטרטגיית הארגון 308
האחריות לביצוע הפרויקט 308
התחל בפרויקט ניסוי (Pilot Project) 308
פרויקט ארגוני 309
אל תזלזל במאמץ מיפוי נתונים והעברת הנתונים........ 309
טפל בבעיות אמינות הנתונים החל מהשלב הראשון 309
הייה מציאותי בכל הקשור לבעיות סנכרון הנתונים בתוך מחסן הנתונים 310
בחר בתבונה את כלי הקצה........ 310
בנה הערכה ריאלית של כוח האדם הנדרש 310
הגדר את כללי אבטחת המידע 311
צפה שעלויות הקמת מחסן הנתונים תהיינה גבוהות מהמתוכנן 311
בנה קטלוג נתונים טוב 311
למד את המשתמשים המתוחכמים ואת מנתחי המידע
כיצד לבנות שאילתות טובות........ 312
סיכום 312

נספח א׳: מילון מונחים 313

נספח ב׳: ביבליוגרפיה 335

הקדמה

מחסן הנתונים (**Data Warehouse**) הפך בשנים האחרונות לאחד הנושאים החמים ביותר בתחום טכנולוגיות המידע. על פי סקרים שונים עולה שמספר הארגונים שעוסקים בצורה כלשהי בהקמת מחסני נתונים, צמח במהירות מ- 10% בשנת 1994 עד ל- 90% בשנת 1998. תוך פרק זמן קצר יחסית הפכו מחסני הנתונים לפתרון המועדף לבניית סביבת מחשוב ייעודית לניהול ואספקת המידע הדרוש לניתוח נתונים ולתמיכה בתהליכי קבלת החלטות עיסקיות. ההתפתחות המהירה של תפיסת מחסן הנתונים באה לאחר שנים רבות בהם ארגונים פיתחו ויישמו מערכות תומכות החלטה (DSS) לצד המערכות התפעוליות, אולם ללא הצלחה משמעותית ועם הרבה בעיות וקשיים. רק בשנים האחרונות הבשילו התנאים להצלחת מערכות תומכות החלטה: הלחצים הבלתי פוסקים על הארגון לקבל החלטות מורכבות ומהירות מחד, ומאידך - ההתפתחויות בטכנולוגיות המידע בכל הקשור לחומרה, תוכנה וכלים מתקדמים לניתוח הנתונים. תנאים אלה הביאו להתפתחות תפיסה כוללת לפיתוח ויישום סביבת מחשוב ייעודית לאספקת מידע לתמיכה בהחלטות - תפיסת **מחסן הנתונים**.

הגדרת המושג "מחסן נתונים" עמומה במידת מה. זו סביבת מחשוב ייעודית לניתוח ולעיבוד אנליטי של הנתונים והפיכתם ל**מידע תומך החלטות**. מחסן הנתונים מורכב משלושה רכיבים עיקריים: רכיב העוסק בתהליכי גזירת הנתונים ממערכות תפעוליות, רכיב העוסק בניהול הנתונים ורכיב העוסק באספקת הנתונים ובביצוע ניתוח המידע. רק לאחר שמתחילים לעכל את משמעות המושג מחסן נתונים, להעמיק ולנתח את מלוא המשמעות של תפיסה זו, מתחילה להתבהר התמונה האמיתית. לפנינו,

- ❖ סביבה עתירת טכנולוגיות מידע שונות,
- ❖ סביבה שצריכה לענות לדרישות משתנות שקשה להגדירן מראש באופן מפורש,
- ❖ סביבה שצריכה לנהל היקף נתונים גדול,
- ❖ סביבה שמכניסה מימד בלתי ידוע עד כה של כפילות נתונים אל הארגון,
- ❖ סביבה העוסקת במיגוון תהליכים, כמו גזירת נתונים, טיוב הנתונים וטעינתם, ועוד.

כמו כל להיט ומגמה טכנולוגית חדשנית, גם מחסן הנתונים מלווה בסיסמאות רבות ובציפיות שחלקן אינן ברות השגה. יש חשיבות רבה בהבנת המציאות והאתגרים האמיתיים שהארגון יצטרך להתמודד איתם בעת יישום מחסן נתונים וללמוד את התועלות שהוא צופה להשיג. יחד עם התלהבות הארגונים באימוץ תפיסת מחסן הנתונים, שסוף כל סוף מאפשרת להפוך את הנתונים האגורים במערכות המידע השונות של הארגון למידע תומך החלטות, באה המציאות ומוכיחה שתפיסת מחסן הנתונים היא תפיסה מורכבת ויישומה דורש משאבים רבים. לארגונים התבררו מספר עובדות:

- ❖ מחסן נתונים לא קונים אלא בונים,
- ❖ מעט מאוד ספקים מסוגלים לספק את הקשת הרחבה של המוצרים הדרושים להקמת מחסן הנתונים,
- ❖ תהליך הקמת מחסן נתונים הוא תהליך מורכב יחסית של שילוב מערכות,
- ❖ הנתונים במערכות התפעוליות אינם במבנה וברמת ניקיון מתאימה לתמיכה בהחלטות,
- ❖ הקמת מחסן נתונים הוא פרויקט ארגוני שנמשך זמן,
- ❖ הפרויקט צורך משאבים רבים והוא בעל מאפיינים שונים מתהליך הקמת מערכת מידע חדשה.

במקביל לסיפורי ההצלחה החלו להישמע גם לא מעט סיפורים על כשלונות ופרויקטים שננטשו לאחר שהושקעו משאבים רבים בהקמתם. בדיעבד הסתבר שחלק מהארגונים קפצו על העגלה במהירות רבה מדי והתייחסו אל פרויקט מחסן נתונים כאל פרויקט נוסף לפיתוח מערכת מידע. הם לא טרחו ללמוד ולנסות להבין את מלוא המשמעויות של הקמת מחסן נתונים ובסופו של דבר נכשלו ביישום נושא חשוב זה והפסידו הרבה זמן וכסף.

למרות האתגרים והקשיים בהקמת מחסן הנתונים, ארגונים חייבים להבין את התועלות שבתפיסת מחסן הנתונים ואת המשמעויות האמיתיות של הקמת סביבת מחשוב חדשה זו. ארגונים שיבינו את התועלות, גם יצליחו להמשיך לתפקד ולהצליח בסביבה העסקית המודרנית, התחרותית והדינמית כל כך. ארגונים מסוימים עלולים להיבהל מהאתגרים ולהתמקד בעיקר במערכות התפעוליות, ולא במערכות תומכות החלטה. גישה כזו מוליכה לסיכון של איבוד אחד היתרונות התחרותיים החשובים ביותר: היכולת לנצל את הנתונים והמידע כמשאב תחרותי. כאשר מסתכלים על הנתונים כמשאב תחרותי, אפשר להבין טוב יותר ומהר יותר מה קורה לארגון, מה קורה למוצרים שהוא מוכר, מה לקוחותיו באמת רוצים וכיצד ניתן להגביר את נאמנותם למוצרי החברה, מה קורה לשוק שבו הארגון פועל ובסופו של דבר - כיצד לקבל החלטות עסקיות טובות יותר.

מטרת ספר זה להציג ולסקור את תפיסת מחסן הנתונים על כל היבטיה: מדוע המערכות התפעוליות ומערכות לתמיכה בהחלטות שונות זו מזו ומדוע צריך להקים סביבת מחשוב ייעודית. הקמת סביבת עיבוד חדשה זו מחייבת:

- ❖ להכיר את הארכיטקטורות השונות של מחסני הנתונים,
- ❖ להכיר את מיגוון השיטות השונות לניתוח נתונים והכלים האנליטיים הפועלים בסביבה זו,
- ❖ להכיר את טכניקות העיצוב של בסיסי הנתונים המשרתים את מחסן הנתונים,
- ❖ לבחון את התמונה הכוללת של התהליכים הקיימים בסביבת מחסן הנתונים,
- ❖ ללמוד על החלופות השונות לפיתוח מחסני נתונים ואת מתודולוגיות היישום שלהם.

הכרת מכלול נושאים מקיף זה תאפשר לכל ארגון להבין בצורה טובה את מלוא המשמעויות של תהליך הקמת מחסן הנתונים, ולקבל את ההחלטות בהתבסס על הבנה מוצקה הן של התועלות והן של הקשיים.

הגישה של הספר היא מעשית ולא תיאורטית. הנושאים מוצגים בלשון פשוטה ולא טכנית או פורמלית, הטקסט מלווה במספר רב של דוגמאות ושל תרשימים. נקודת המוצא היא שהעקרונות והארכיטקטורה של מחסן הנתונים חשובים בהרבה מהמוצרים הספציפיים המרכיבים אותו. על כן, חלק מהמוצרים מוזכרים באופן כללי וכמעט ללא פירוט. טכנולוגיית מחסני הנתונים חדשה יחסית ומורכבת מאוסף גדול של מוצרים שונים המשתנים כל הזמן. לכן יש חשיבות רבה יותר להבנת העקרונות, התמונה הכוללת והמקום של כל מוצר במסגרת הארכיטקטורה, מאשר לאופן הפעולה הספציפי של מוצר כלשהו. בנוסף, ככל שהזמן שעובר והספר יהיה מונח על שולחן עבודתך, הפרטים לגבי מוצר זה או אחר כבר לא יהיו תקפים, בשעה שהעקרונות יישארו איתנו זמן רב. הנקודה העיקרית המודגשת בספר היא החשיבות בבניית מחסן נתונים גמיש המסתגל בקלות יחסית לשינויים שהארגון עובר, לדרישות המידע החדשות, לטיפול בנפחי נתונים הולכים וגדלים ולמוצרים החדשים. חוסר הבנה של התהליכים העיקריים של סביבת מחסן הנתונים והתבססות על ארכיטקטורה שלא הושקעה בה המחשבה הנדרשת, עלולים לגרום לכישלון התפיסה ולעגמת נפש, גם אם בשלב ההקמה הארגון השתמש במוצרים מהשורה הראשונה.

הספר התגבש כתוצאה מפעילותי בתחום טכנולוגיות המידע, ובעיקר בנושאי בסיסי נתונים, טכנולוגיות שרת/לקוח ומערכות מוכוונות אובייקטים. בשנים האחרונות נחשפתי לנושא מחסני נתונים, נתתי מספר רב של הרצאות במסגרות שונות של מוסדות הוראה וימי עיון מקצועיים שונים ועסקתי בייעוץ להקמת מערכות כאלו. זו אחת מטכנולוגיות המידע החשובות ביותר ויחד עם זאת אחת המורכבות למימוש ויישום, ולכן אני מאמין בחשיבות ספר בשפה העברית שיפרוס בפני הקורא את כל רוחב היריעה של הנושא.

מבנה הספר

נסקור בקצרה את תוכן הפרקים השונים של הספר, כדי לאפשר לקורא לקבל התרשמות ראשונית ומהירה של תכולת הספר.

❖ **פרק 1 - רקע ומושגי יסוד**

פרק זה מציג את הרקע להתפתחות תפיסת מחסן הנתונים תוך הדגשת שתי המגמות העיקריות. מצד אחד, השינויים בסביבה העסקית שהביאה ללחצים מתמידים לקבלת החלטות מהירות ומבוססות; ומצד שני, ההתפתחויות המהירות בטכנולוגיית המידע עם הופעת המחשבים האישיים, השרתים המקביליים רבי העצמה והזולים יחסית, בסיסי הנתונים היחסיים המסוגלים לטפל בהיקפי נתונים גדולים והופעת רשתות התקשורת המהירות. הפרק מפרט את ההבדלים המהותיים בין היישומים התפעוליים של הארגון התומכים בתהליכי העבודה השוטפים לבין היישומים האנליטיים העוסקים בניתוח מידע ותמיכה בתהליכי קבלת ההחלטות. לסיום, מוצגת ההתפתחות רבת השנים של המערכות לקבלת החלטות, שהחלה בשנים הראשונות של המחשוב והשתכללה עם הזמן עד להגדרת התפיסה המודרנית של מחסן הנתונים.

❖ **פרק 2 - ארכיטקטורת מחסן הנתונים (Data Warehouse Architecture)**

פרק זה סוקר את הארכיטקטורות השונות למימוש מחסן הנתונים: מחסן הנתונים הארגוני; מרכול הנתונים; מחסן הנתונים הרב-שכבתי המשלב את היתרונות של המחסן הארגוני עם היתרונות של מרכולי הנתונים; ומאגר הנתונים התפעולי שהוא סוג מסוים של מחסן נתונים המתאים לאספקת מידע לתהליכים תפעוליים.

❖ **פרק 3 - סקירת מרחב הטכנולוגיות לניתוח מידע (Analytical Processing Technology Space)**

פרק זה מציג את כל המרחב של נושא ניתוח המידע והעיבודים האנליטיים לתמיכה בקבלת החלטות. כדי להבהיר את תהליך ניתוח המידע, מוצגת שרשרת הפעילויות ההופכת נתונים למידע, את המידע לידע ואת הידע להחלטות עסקיות. המרחב המלא של שיטות ניתוח מידע שונות מוצג גם הוא: מחוללי שאילתות ודוחות המשמשים לניתוחים פשוטים יחסית, כלי OLAP המורכבים יותר והמאפשרים ניתוח רב-מימדי והכלים לכריית נתונים המאפשרים לבצע ניתוחי מידע מורכבים מאוד ולגלות חוקיות ותבניות בלתי ידועות בנתונים. הפרק מסתיים בהצגת מפה של מרחב הטכנולוגיות השונות לניתוח מידע על פי דרישות המידע וסוגי משימות הניתוח הנדרשים.

❖ **פרק 4 - ניתוח מידע רב-מימדי (Multi Dimensional Analysis)**

פרק זה מעמיק ומסביר את הדרכים לניתוח רב-מימדי של מידע, אחת מפעילויות הניתוח והעיבוד האנליטי הנפוצות ביותר. הפרק עוסק בסקירת מושגי היסוד בניתוח רב-מימדי: מה זה מימד, מה זו עובדה, מהן ההיררכיות בתוך המימדים, מהן הפעולות העיקריות בניתוח רב-מימדי, מהי הקוביה הרב-מימדית ועוד.

❖ **פרק 5 - ארכיטקטורות של מוצרים לניתוח רב-מימדי (OLAP Architectures)**

הפופולריות הרבה של שיטת הניתוח הרב-מימדי תרמה לפיתוח מספר רב של מוצרים התומכים בה. פרק זה סוקר את הארכיטקטורות השונות למימוש הניתוח הרב-מימדי ובאופן מיוחד את שתי הארכיטקטורות העיקריות: ארכיטקטורת **MOLAP** המבוססת על בסיס נתונים רב-מימדי ייעודי, וארכיטקטורת **ROLAP** המבוססת על מנוע רב-מימדי המבצע את מירב העיבודים הדרושים אולם מנהל את הנתונים בבסיס נתונים טבלאי רגיל. הפרק סוקר את ההבדלים בצורת המימוש של ארכיטקטורות אלו ומציג את המסקנה שהן אינן נוגדות, כמו שלעיתים יצרנים היו רוצים לגרום לנו לחשוב, אלא משלימות.

❖ **פרק 6 - עיצוב רב-מימדי של בסיסי נתונים טבלאיים (Multi Dimensional Design of Relational Databases)**

פרק זה סוקר את שיטת העיצוב המיוחדת של בסיסי הנתונים הטבלאיים הפועלים בסביבת מחסן הנתונים. חלק גדול מתפיסות העיצוב הרגילות של בסיסי נתונים טבלאיים הפועלים בסביבות של עיבוד תנועות אינו מתאים לייעוד מחסן הנתונים. הפרק מציג את תפיסת העיצוב המיוחדת של בסיסי נתונים המשרתים מחסן נתונים תוך הצגת מושגי היסוד: טבלת עובדות, טבלת מימד, טבלאות לא מנורמלות, סכמות כוכב, סכמות פתיתי שלג, טבלאות סיכומים, שיטות לניהול שינויים במימדים ועוד.

❖ **פרק 7 - כריית נתונים (Data Mining)**

פרק זה מציג את אחד הנושאים ה'חמים' ביותר כיום במסגרת תפיסת מחסן הנתונים. לרבים נושא זה נשמע כמו מדע בדיוני שרלוונטי בעיקר לחוקרים ומדענים. המציאות כמובן שונה מאוד. כריית הנתונים מורכבת ממספר טכנולוגיות מחשוב ומוצרי תוכנה שונים המאפשרים לארגון לנתח את הנתונים, לגלות חוקיות פנימית ובלתי ידועה מראש, לגלות תבניות ולבצע תחזיות מבוססות. לכריית נתונים משתמשים במספר טכניקות סטטיסטיות וטכניקות הלקוחות מתחומי מחקר מתקדמים כמו אינטליגנציה מלאכותית, מערכות לומדות וכד'. הפרק סוקר את מיגוון היישומים האפשריים, את הטכניקות הנפוצות לכריית נתונים ואת מקומו של תהליך כריית הנתונים במסגרת תפיסת מחסן הנתונים.

❖ **פרק 8 - הטכנולוגיה המקבילית בשירות מחסן הנתונים (Parallel Technology in Data Warehouses)**

ההתפתחויות המהירות בשנים האחרונות בכל הקשור לארכיטקטורות חומרה מרובות המעבדים השפיעו רבות על תפיסת מחסן הנתונים. התברר שניתן לנהל ולעבד באופן יעיל נפחי נתונים עצומים וגם לעשות זאת בעלויות סבירות. לטכנולוגיה המקבילית היתה השפעה על מספר רב מאוד של תחומים הקשורים למחסן הנתונים: ניהול בסיס נתונים, אופטימיזציה של גישה לבסיס נתונים, טעינה מהירה של נתונים ועוד. הפרק סוקר את הארכיטקטורות העיקריות של ריבוי מעבדים, את השיטות לעיבוד מקבילי של שאילתות ואת שיטת חלוקת בסיס הנתונים למחיצות כשיטה לניהול יעיל של נפחי נתונים גדולים.

❖ **פרק 9 - סקירת התהליכים העיקריים במחסן הנתונים (Main Processes in Data Warehouses)**

מחסן הנתונים הינו עתיר תהליכי עבודה: תהליכי גזירת הנתונים מהמערכות התפעוליות, תהליכי סיכום נתונים לייעול ניתוח הנתונים, תהליכי אספקת הנתונים לכלי הניתוח השונים ותהליכי גיבוי והעברת הנתונים למדיה זולה יותר. המציאות היא, שארגון משקיע אנרגיה רבה בהקמת ותחזוקה שוטפת של תהליכים אלה. פרק זה סוקר את מכלול התהליכים תוך מתן דגש מיוחד לתהליכים המורכבים לגזירה, לטיוב ולהמרה של הנתונים, שנעשים לרוב באמצעות תוכנה ייעודית.

❖ **פרק 10 - מידע על נתונים (Metadata)**

פרק זה מציג את נושא המידע על הנתונים. נושא זה הינו בעל חשיבות רבה להצלחת מחסן הנתונים, אולם בגלל מורכבותו והעדר כלים מספיק טובים ומקיפים, הוא אינו זוכה לתשומת הלב המתאימה מצד מפתחי מחסן הנתונים. הפרק סוקר את הסוגים הרבים והמגוונים של מידע על הנתונים שיש לנהל בסביבת מחסן נתונים: משמעות הנתונים, מבנה הנתונים, היכן הם מאוחסנים, למתי הם נכונים, מה מקורם ועוד.

❖ **פרק 11 - מרכולי נתונים (Data Marts)**

מרכול הנתונים הפך בשנים האחרונות לאחד הנושאים החשובים והבולטים ביותר במסגרת תפיסת מחסן הנתונים. למעשה, זהו מחסן נתונים ממוקד במחלקה מסוימת, בנושא מסוים או במטרה מסוימת. תפיסה זו התפתחה על רקע הקשיים בהקמת ומימוש תפיסת מחסן הנתונים הארגוני, שהינה מקיפה ורחבה. מרכול הנתונים אינו מתיימר לתת פתרון מקיף לארגון בתחום התמיכה בקבלת החלטות. זהו גם מקור ההתלהבות והעניין בו: קל יחסית להקימו, עלות ההקמה זולה יותר, הבעלות עליו ברורה, קל יותר להצדיקו מבחינת עלות/תועלת. אך בצד התועלות הברורות של תפיסת מרכול הנתונים, יש לא מעט סיכונים בתפיסה זו. פרק זה סוקר את תפיסת מרכול הנתונים, מה הרקע לפופולריות שלו, מהם הסיכונים במרכולי נתונים, מה הארכיטקטורה שלהם ומהם המודלים השונים להקמת מרכולי נתונים.

❖ **פרק 12 - מתודולוגיה ליישום מחסן הנתונים (Data Warehouse Implementation Methodology)**

הפרק האחרון של הספר מוקדש למתודולוגיית היישום של מחסני נתונים. הפרק בוחן מדוע יש צורך במתודולוגיה ייחודית ומותאמת, מהם עקרונות המתודולוגיה, מהם השלבים השונים בתהליך הקמת מחסן הנתונים, מי הם בעלי התפקידים השונים הפועלים בסביבת מחסן הנתונים. לסיכום ניתנות מספר עצות מעשיות לבניית מחסן הנתונים.

❖ **נספחים -** שני נספחים בספר זה. האחד עוסק במינוחים והוא משמש גם כאינדקס, והשני מפרט את הביבליוגרפיה על פיה נערך ספר זה.

קהל היעד של הספר

הספר מיועד לקהל קוראים מגוון, וביניהם:

- ❖ **מנהלים ומשתמשים עתידיים של מחסן הנתונים.** חשוב שציפיותיהם ממחסן הנתונים תהיינה מעשיות וניתנות למימוש, ורצוי שיהיו מסוגלים להגדיר את דרישותיהם מתוך הבנת התהליך הכולל לבניית מחסן הנתונים.
- ❖ **מנהלי מחשוב** המבקשים להקים מחסן נתונים בארגון. אלה מבקשים להבין טוב יותר את מלוא המשמעויות של פרויקט ההקמה והיישום ובכלל זה: מהו מחזור החיים של הפרויקט ובמה הוא שונה ממחזור חיים של מערכת מידע רגילה, מה המתודולוגיה להקמה ויישום, אילו סוגי עובדים צריך להקצות, מה האתגרים הטכנולוגיים שהם צפויים להתמודד איתם, ועוד.
- ❖ **מנתחי מערכות** המבקשים להבין את הטכנולוגיה החדשה ולהצטרף לפרויקט מחסן נתונים. עליהם ללמוד על תהליכי ההקמה, המושגים העיקריים, שיטת העבודה לניתוח המערכת, הכלים הדרושים ועוד.
- ❖ **מעצבי בסיסי נתונים** הרגילים לעצב בסיסי נתונים למערכות עיבוד תנועות רגילות ומבקשים להבין מה ההבדל בין טכניקות העיצוב הרגילות לבין טכניקות העיצוב המיוחדות לסביבת מחסן הנתונים.
- ❖ **ארכיטקטים של מערכות מידע** המבקשים לשלב את תפיסת מחסן הנתונים במסגרת הארכיטקטורה הכוללת של מערכות המידע בארגונים.

הטבלה הבאה מציגה את הפרקים השונים המומלצים ביותר לסוגי הקוראים השונים.

שם פרק	מנהלים/ משתמשים	מנהלי מחשוב	מנתחי מערכות	מעצבי בסיסי נתונים	מעצבי מערכות מידע
רקע ומושגי יסוד	✓	✓	✓	✓	✓
ארכיטקטורת מחסן הנתונים	✓	✓	✓	✓	✓
סקירת מרחב הטכנולוגיות לניתוח מידע	✓	✓	✓		✓
ניתוח מידע רב-מימדי	✓	✓	✓		✓
ארכיטקטורות של מוצרים לניתוח רב-מימדי		✓	✓	✓	✓
עיצוב רב-מימדי של בסיסי נתונים טבלאיים				✓	
כריית נתונים	✓	✓	✓		
הטכנולוגיה המקבילית בשירות מחסן הנתונים		✓		✓	
סקירת התהליכים העיקריים במחסן הנתונים	✓	✓	✓		
מידע על הנתונים	✓	✓	✓		
מרכולי נתונים	✓	✓	✓		✓
מתודולוגיה ליישום מחסן נתונים		✓	✓		

פרק 1: רקע ומושגי יסוד

מבוא

יכולתם של ארגונים הפועלים בסביבה העסקית המודרנית לקבל החלטות אפקטיביות ובעיתוי הנכון, הופכת לקריטית יותר ויותר. הארגונים עוברים תהליכי שינוי מתמידים, הן בתהליכי העבודה והן במבנים הארגוניים. שינוי זה הוא תוצאה של גיוון המוצרים והשירותים שהם מציעים ללקוחותיהם, קיצור אורך החיים של המוצרים שהם מייצרים, תהליכי רכישות ומיזוגים, תהליכי הפרטה במגזרי שוק ההולכים וגדלים וניסיון לנצל הזדמנויות עסקיות חדשות בשווקים המקומיים והעולמיים. הסתיים העידן בו ארגון יצר לעצמו יתרון תחרותי באופן חד-פעמי ואחר כך נהנה ממנו למשך שנים רבות. כיום נדרש הארגון להתחדש כל הזמן, לזהות את המגמות המשתנות במהירות, לזהות בזמן את המתחרים החדשים, לזהות את השינויים הצפויים בכללי המשחק וליצור את היתרון התחרותי פעם אחר פעם.

האתגר העומד בפני המנהלים, ברמות הניהול השונות, הוא לקבל החלטות עסקיות נכונות המשפיעות באופן חיובי על "השורה התחתונה" - ההצלחה העסקית. אתגר זה הופך למורכב יותר, והמחיר של החלטה שגויה הופך להיות יקר יותר. מקבלי ההחלטות נדרשים לענות על מיגוון שאלות בעלות השלכה משמעותית על יכולת הארגונים לשרוד ולשמור על מעמדם התחרותי.

המבנים הארגוניים החדשים 'שטוחים' יותר ובעלי מספר קטן יותר של רמות היררכיה. הארגונים הפקידו את הצלחתם העסקית בידי מספר הולך וגדל של מנהלים העוסקים בתהליכי קבלת החלטות באופן פעיל ושוטף. ניתן לומר שבסביבה התחרותית המודרנית, תלויה הצלחת הארגון ביכולתם של מקבלי ההחלטות להבין ולפענח את הזרם הבלתי פוסק של הנתונים ושל המידע. יכולות נוספות אותן צריכים מקבלי ההחלטות להציג הן התבוננות על הנתונים מנקודות מבט שונות, איתור מהיר של מצבים חריגים ובחינת תסריטים אפשריים שונים. עליהם להגיע להחלטות העסקיות במהירות וביעילות.

הלחצים הבלתי פוסקים לקבלת החלטות מורכבות ומהירות המבוססות על מידע אמין, הביאו לצמיחת מערכות מחשוב ייעודיות, **מערכות תומכות החלטות** (Decision Support Systems). כיום, יותר מאי-פעם, הפכו המנהלים לתלויים במידה רבה ביכולות של מערכות אלו לסייע להם בהפיכת זרם הנתונים לידע ולהחלטות עסקיות, שיבטיחו לארגון יתרון תחרותי.



תרשים 1.1: החלטות, החלטות.

במשך השנים הופנו עיקר המאמצים והמשאבים של הארגונים לבנייה ויישום של מערכות מחשוב תפעוליות שעיקר מטרתן היתה לתמוך, לייעל ולשפר את תהליכי העבודה השוטפים. דוגמאות למערכות מסוג זה הן מערכות להזמנת כרטיסי טיסה, מערכות מחשוב סניפי בנק, מערכות ניהול קשרי לקוחות, מערכות חיוב וגבייה עבור חברות תשתית (כגון חברות טלקומוניקציה חשמל ומים), מערכות פיננסיות כוללניות, מערכות מחשוב בנקודות מכירה ברשתות קמעוניות, מערכות כוללניות לתכנון ניהול ותפעול משאבי הארגון ונוספות.

יחד עם התחכום ההולך וגדל של מערכות תפעוליות אלו, הגיעו רוב הארגונים למסקנה שהשגת היתרון התחרותי בסביבה העסקית המודרנית תלוי פחות בתהליכי הייעול והאופטימיזציה של המערכות העוסקות בצד התפעולי ויותר - ביכולת הארגון לעשות שימוש יעיל ונכון במידע ובנתונים שמערכות תפעוליות אלו מעבדות ומנהלות לאורך השנים. הארגונים החלו לזהות את הפוטנציאל העצום הגלום בשיפור תהליכי קבלת החלטות ובצורך לצייד את מקבלי ההחלטות במידע אמין ומקיף. הבעיה העיקרית היא שתהליכי קבלת ההחלטות והמידע הנדרש עבורם, שונים באופן מהותי מהתהליכים התפעוליים והמידע הנדרש עבורם. המערכות התפעוליות לא נבנו ועוצבו באופן שיאפשר תמיכה בתהליכי קבלת ההחלטות של הדרגים הניהוליים. על רקע זה החל להתפתח זן חדש של מערכות מידע, שעיקר ייעודן ומיקודן הוא **אספקת מידע לתמיכה בתהליכי קבלת החלטות.**

מערכות תומכות החלטה התפתחו במהלך 15 השנים האחרונות. בתחילה היו אלו מערכות לא גמישות המבוססות על מחשבים מרכזיים. אחר כך הגיע תור המערכות המבוססות על גיליונות אלקטרוניים, הפועלים במחשבים אישיים המנותקים ממערכות המחשוב הארגוניות. השלב הבא היה פיתוח מערכות המיועדות למנהלים בכירים בלבד (EIS). הגירסה האחרונה של מערכות אלו הן מערכות תומכות החלטה מתקדמות, המבוססות על טכנולוגיות מידע מודרניות ופועלות במסגרת תפיסה חדשה ורחבה יותר - **מערכות מחסן הנתונים** (Data Warehouse). רוב מנתחי השוק צופים שחלק ניכר מהשקעות הארגונים בטכנולוגיית המידע בשנים הקרובות יופנה למערכות השייכות לקטגוריה זו.

במסגרת **מחסן הנתונים** מנוהלים הנתונים הנדרשים לתמיכה בתהליכי קבלת ההחלטות, בבסיס נתונים אינטגרטיבי ובמבנה התואם את הדרישות הייחודיות של היישומים לתמיכה בקבלת החלטות. המערכות התפעוליות בארגון בנויות ומעוצבות באופן שיאפשר מחשוב ואוטומציה של מירב הפעולות היומיומיות והשוטפות של הארגון; ולעומתן, המערכות לתמיכה בקבלת החלטות צריכות להיבנות מתוך מטרה לסייע למנהלים לבסס את תהליכי קבלת ההחלטות שלהם על מידע אמין, המאורגן באופן ייעודי למטרה זו. המערכות התפעוליות טובות מאוד בקליטת אירועים וניהול הנתונים באופן יעיל ואמין, אבל הן אינן עונות על הדרישות בכל הקשור לאספקת מידע תומך לתהליכי קבלת החלטות. כפי שחלק מהמנהלים מתבטאים לעיתים - **יותר מדי נתונים ופחות מדי מידע**. כתוצאה מהצורך הבלתי פוסק במידע לקבלת החלטות, וחוסר ההתאמה של המערכות התפעוליות לספק מידע זה, החלה הצמיחה המהירה של **מערכות תומכות החלטה** המבוססות על תפיסת מחסן הנתונים.

אין חולקים על כך שתפיסת מחסן הנתונים הפכה בשנים האחרונות לאחד הנושאים החשובים והמדוברים ביותר בתחום טכנולוגיית המידע. תוך פרק זמן קצר יחסית הפכו מחסני הנתונים לפתרון המועדף לבניית סביבת עבודה מיוחדת ומותאמת לאספקת מידע (Information Delivery Environment) הנדרש לתמיכה בתהליכי קבלת החלטות וניתוח הנתונים בארגון העסקי המודרני. מספר הארגונים העוסקים בהקמת מחסני הנתונים גדל והולך. מולם קיים מספר רב של ספקי כלי תוכנה הבונים כלים ופתרונות המיועדים לשוק מחסני הנתונים. בנוסף לאלה, הספרות המקצועית מלאה בסקירות ומאמרים בנושא זה. מחקרי שוק שונים מצביעים על כך שרוב הארגונים הגדולים עוסקים באופן פעיל בתהליכי הקמת מחסני נתונים, ואלה שעדיין לא הקימו אותם מתכוונים לעשות זאת בעתיד הקרוב. הארגונים משקיעים משאבים ניכרים - אנשים, טכנולוגיה וכסף - כדי לבנות ולהשתמש במחסני הנתונים.

המטרה העיקרית במאמצים אלה היא לאפשר נגישות קלה לנתונים שהוכנו מראש עבור יישומים ייעודיים לתמיכה בתהליכי קבלת החלטות: דיווחים ניהוליים, שאילתות מתוחכמות, ניתוחים רב-מימדיים (Multi Dimensional Analysis), סימולציות What-If, מערכות מידע למנהלים (Executive Information Systems), ניתוחים סטטיסטיים וכריית נתונים (Data Mining). קיימת הסכמה רחבה, שתפיסת מחסן הנתונים היא התפיסה הנכונה והמקיפה לבניית תשתית ניהול הנתונים הנדרשים בתהליכי קבלת ההחלטות. תפיסת מחסן הנתונים מהווה תשתית מוצקה למיגוון יישומים אלה.

תפיסת מחסן הנתונים, כתשתית עיקרית לכלל המערכות תומכות ההחלטה בארגון, התגבשה בשנים האחרונות והתפתחה כתוצאה משתי מגמות עיקריות:

❖ **השינויים בסביבה העסקית:** על ארגון מודרני הפועל בסביבה עסקית תחרותית מופעלים לחצים בלתי פוסקים לאתר שווקים והזדמנויות חדשות, לשפר את איכות השירות ללקוחות, להוזיל את עלות המוצר, לקצר את זמן הבאת המוצרים החדשים לשוק (Time to Market) ולשפר את תהליכי קבלת ההחלטות ובקיצור - להתייעל. דרישות אלו הביאו ללחצים מצד מקבלי ההחלטות בארגון לשיפור איכות וזמינות המידע העומד לרשותם והנדרש על ידם עבור תהליכי קבלת ההחלטות.

❖ **השינויים בטכנולוגיית המידע:** ההתפתחויות בטכנולוגיית המידע בכלל, ומהפיכת המחשב האישי בפרט, הביאו לכך שניתן כיום לבנות בסיסי נתונים גדולים מאוד המנוהלים בשרתים מרובי מעבדים. בשרתים אלה ניתן לנהל נפחים עצומים של נתונים בצורה יעילה וגמישה בפלטפורמת מחשוב נפרדת, ללא פגיעה במערכות התפעוליות של הארגון. עלויות החומרה ואחסון הנתונים נמצאות במגמה בלתי פוסקת של ירידה, במקביל לעלייה המתמדת בביצועים ובנפחי האחסון האפשריים. בנוסף, חדירת המחשבים האישיים לכל דרגי הארגון הביאה לשיפור היכולות האישיות של המשתמשים, בכל הקשור למחשוב וליכולות הצגת וניתוח נתונים. שיפור זה הוא תוצאה של ממשקי המשתמש הגרפיים והופעת כלי שאילתות וניתוח נתונים מתוחכמים, כלי ניתוח רב-מימדיים וכדומה.

תפיסת מחסן הנתונים היא תפיסה רחבה המנסה לספק פתרון קצה לקצה (end-to-end solution) בכל ההיבטים הקשורים לנושא, וביניהם: הארכיטקטורה הכוללת של מחסן הנתונים, תהליכי העיצוב הייחודיים והשונים מתהליכי העיצוב של יישומים תפעוליים רגילים, כלי התוכנה הייעודיים לנושא תמיכה בקבלת החלטות וניתוח נתונים, דרכים ליישום תפיסה זו בארגון, ועוד.

למען האמת, מי שמתבונן בתהליכים העוברים על הארגון העסקי בכלל, ועל טכנולוגיית המידע בפרט, אינו צריך להיות מופתע מהצמיחה המהירה ומתשומת הלב הרבה שתפיסת מחסן הנתונים זכתה לה. המציאות, ברוב הארגונים המודרניים, היא שהנתונים מפוזרים במספר רב של יישומים תפעוליים שונים הפועלים על פלטפורמות מחשוב שונות (חומרה, מערכות הפעלה, רשתות תקשורת, מערכות לניהול בסיסי נתונים, שפות תכנות וכדומה). עובדה זו גרמה לכך שהנגישות לנתונים הפכה למורכבת, מסובכת ויקרה מאוד, במונחי המשאבים והזמן הנדרשים לשם כך. על קרקע זו צמחה ההבנה הברורה שהפתרון יכול לבוא רק מבנייה של סביבה מחשובית נפרדת, המותאמת במיוחד ליעדיה - סביבה לאספקת מידע לתמיכה בקבלת החלטות. על רקע החשיבות ההולכת וגוברת לשיפור תהליכי קבלת החלטות בסביבה העסקית המודרנית, ברורה לחלוטין המרכזיות והחשיבות שתפיסת מחסן הנתונים הולכת ותופסת על במת המחשוב המודרנית.

המגמות האסטרטגיות העיקריות בטכנולוגיית המידע

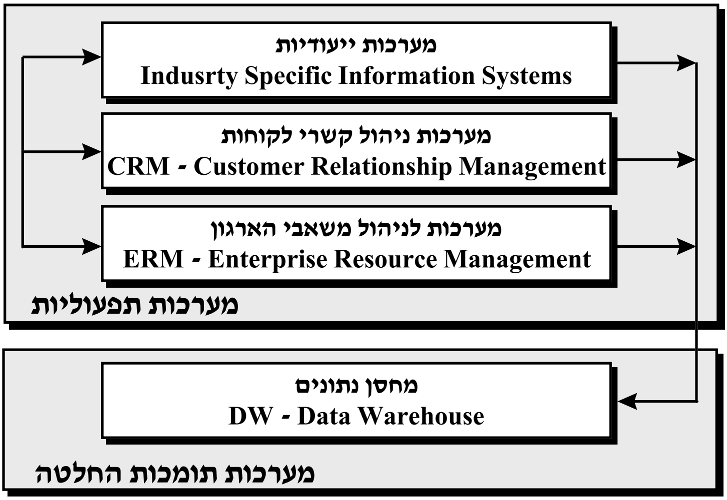
טכנולוגיית המידע עברה (ועוברת) תהפוכות רבות בשנים האחרונות. כדי להדגיש ולמקד את חשיבות נושא מחסן הנתונים, נסקור בקצרה את המגמות העיקריות בטכנולוגיית המידע, ונצביע על כך שנושא מחסן הנתונים הוא אחד הנושאים המרכזיים בתהליכי השינוי שהיא עוברת. נתבונן במגמות ובשינויים בטכנולוגיית המידע משתי נקודות מבט שונות - מנקודת מבט של מבנה וסוגי היישומים בהם משתמש הארגון ומנקודת המבט של השינויים בטכנולוגיות המידע. כפי שנראה בהמשך, מחסן הנתונים הוא תולדה של מגמות אלו.

המגמות העיקריות בתחום הארכיטקטורה היישומית

במשך השנים פיתחו הארגונים מערכות מידע שונות אשר שירתו אותם בתחומי הפעילות השונים שלהם. עם השינוי בסביבה העסקית החלו הארגונים לבחון מחדש את הארכיטקטורה הכוללות את היישומים שלהם, והחלו להתאים אותה לסביבה המודרנית בה הם פועלים - סביבה עסקית המאופיינת על ידי תחרות רבה, פעילות גלובלית לצד הפעילות המקומית ודינמיות גבוהה. הארגון העסקי חייב לשפר באופן מתמיד הן את יכולת התגובה שלו לשינויים בשוק והן את יכולתו להיות יעיל יותר בכל הקשור לניהול המשאבים הנדרשים כדי להפוך את התשומות לתפוקות.

ארגון שאינו מבין היטב את לקוחותיו והעדפותיהם, אינו יכול לכוון את מאמציו השיווקיים לקהל המטרה הנכון, אינו מזהה במהירות את השינויים בשוק ואינו מסוגל להגיב במהירות. ארגון כזה ימצא את עצמו מהר מאוד במצב קשה, וסיכוייו לשרוד קטנים והולכים. הסביבה העסקית החדשה מחייבת את הארגון להפוך מארגון **ממוקד מוצר** (Product Oriented) לארגון **ממוקד לקוח** (Customer Oriented). ארגונים אלה חייבים להבין טוב יותר את לקוחותיהם ואת הצרכים הייחודיים שלהם, לספק להם רמת שירות גבוהה יותר ולהבטיח את נאמנותם לאורך זמן. על רקע מגמות אלו, הפכו מערכות המידע של הארגון מכלי תפעולי לכלי אסטרטגי ממדרגה ראשונה, אשר יכול להבטיח במידה רבה את הצלחתו העסקית של הארגון.

השינויים בסביבה העסקית ובטכנולוגיית המידע, הביאו להתבוננות מחודשת גם על הארכיטקטורה הכוללת של מערכות המידע בארגון. ברמת הפשטה מסוימת ניתן לומר שהארכיטקטורה הכוללת של רוב הארגונים תהיה מורכבת מארבע השכבות האלו: מערכות ייעודיות, מערכות ניהול משאבים, מערכות ניהול קשרי לקוח ומערכות תומכות החלטות.



תרשים 1.2: ארכיטקטורה כלל ארגונית עקרונית

תרשים 1.2 מציג את ארבע השכבות העיקריות של ארכיטקטורת-העל ואת קשרי המידע העיקריים ביניהן. נפרט כל אחת משכבות אלו.

❖ **מערכות ייעודיות (Industry Specific Systems):** לכל ארגון יש אוסף של מערכות ייחודיות (Vertical Applications) המותאמות לתחומי עיסוקו. בדרך כלל, המערכות הייעודיות מבטאות את השוני בין הארגונים השונים ואת תחומי העיסוק שלהם, כמו בנקאות, ביטוח, כרטיסי אשראי, טלקומוניקציה, חשמל, בריאות, תעופה, מלונאות, הוראה וכדומה. בקטגוריה זו נמצאות מערכות מידע מסוגים שונים, וביניהן נציין מערכות כגון מערכות חיוב, גבייה וניהול לקוחות (Billing and Customer Care), המיועדות למיגוון רחב של שווקים (כמו טלקומוניקציה, טלוויזיה בכבלים, חברות חשמל, חברות מים וכדומה); מערכות הנדסיות, המטפלות בהיבטים שונים של התשתית ההנדסית של הארגון (כמו תשתית התקשורת והמרכזיות של חברות טלקומוניקציה, תשתית רשת חברות חשמל, תשתית צנרת המים וכדומה); מערכות בנקאיות; מערכות רפואיות ומידע על חולים; מערכות ניהול קופות רושמות ונקודות מכירה ברשתות קמעוניות; מערכות הזמנות עבור חברות תעופה; מערכות ניהול חברות כרטיסי אשראי וכדומה.

בדרך כלל השוני בין הארגונים בא לידי ביטוי במערכות אלו, ולכן מקובל לומר כי אלו הן המערכות האסטרטגיות והקריטיות ביותר בארגון. בדרך כלל המערכות מפותחות על ידי הארגונים עצמם, או מבוססות על חבילות תוכנה העוברות תהליכי התאמה משמעותיים מאוד.

❖ **מערכות ניהול משאבים (ERM – Enterprise Resource Management):** מערכות ניהול משאבים הידועות גם בשם **מערכות ERP** (Enterprise Resource Planning), מהוות כיום נדבך חשוב בארכיטקטורה הכלל ארגונית ברוב הארגונים. מערכות אלו הן פלטפורמה אינטגרטיבית הנותנת מענה כולל לניהול משאבי הארגון - משאבי אנוש, כספים, מכירות, הפצה, לוגיסטיקה (רכש, מחסנים ומלאי), ייצור ותכנון הייצור, ניהול פרויקטים, תמחיר, רכוש קבוע ועוד. בגלל המכנה המשותף הרחב בין הארגונים השונים בכל הקשור לניהול המשאבים שלהם, הגיעו רוב הארגונים להכרה בדבר חוסר הכדאיות בהשקעה בפיתוח עצמי והכירו בכדאיות השימוש בחבילת תוכנה סטנדרטית וגמישה.

מערכות ניהול משאבים נמצאות כיום בהישג ידם של רוב הארגונים. הן מקיפות וגמישות וניתן להתאימן למיגוון הצרכים של הארגונים. למשל, ארגונים יצרניים בעלי תהליכי ייצור שונים, ארגוני שירות כמו חברות טלקומוניקציה, בנקאות, ביטוח, ארגוני בריאות ורבים נוספים. עבור חלק מהארגונים, בעיקר התעשייתיים, מערכות אלו מהוות פתרון כמעט שלם לכל צרכי המחשוב. לעומת זאת, עבור ארגונים אחרים הן רק חלק ממכלול מערכות המחשוב שלהם, ולרוב הן משולבות במערכות הייעודיות. בין המערכות המובילות כיום בעולם בתחום המערכות הכוללות לניהול משאבים ניתן לציין את: R/3 של חברת SAP, Oracle Applications של חברת Oracle, Baan, PeopleSoft, ועוד.

❖ **מערכות ניהול קשרי לקוח (CRM - Customer Relationship Management):** המעבר מארגונים ממוקדי מוצר לארגונים ממוקדי לקוח, תרם לכך שרוב הארגונים מדגישים כיום את ניהול המידע על הלקוחות והמגע עם הלקוחות. מערכות אלו הפכו לאבן בניין חשובה בארכיטקטורה הכוללת של מערכות המידע, לצד המערכות הייעודיות ומערכות ניהול המשאבים. מערכות ניהול קשרי לקוח עוסקות במיגוון נושאים וביניהם: ניהול מכירות ואינטראקציות לקוח (Contact Management), ניהול מוקדי שירות ומכירות ללקוח (Call Center), תמיכה בלקוחות (Help Desk), ניהול מבצעי שיווק ודיוור ישיר (Telemarketing) וכדומה.

מערכות ניהול קשרי לקוח עושות שימוש הולך וגדל בטכנולוגיית Computer) CTI Telephony Integration), המאפשרת שילוב מתוחכם של מערכות מחשוב עם פלטפורמות מיתוג (מרכזיות טלפונים). שילוב זה מאפשר בניית יישומים מתוחכמים, שבהם נמצא ניתוב חכם של שיחות טלפון על סמך נתונים המנוהלים במערכות המידע, שימוש במערכות מענה קולי אינטראקטיבי, שילוב פקס ותמונה וכדומה. בתחום זה עדיין לא ניתן להצביע על מערכת אחת אינטגרטיבית כוללת ולכן, בדרך כלל, הארגון צריך להתבסס על מספר מערכות ולטפל באינטגרציה ביניהן. שים לב ל"תופעה" החדשה, בה מספר גדל והולך של יצרני מערכות לניהול משאבים רוכשים מערכות לניהול קשרי לקוח, ומשלבים אותן במערכות שלהם.

❖ **מערכות תומכות החלטה (Decision Support Systems):** אלו הן מערכות ייעודיות לתמיכה בתהליכי קבלת החלטות, או בשמן המודרני **מחסני נתונים**. מערכות אלו פועלות לצד כל סוגי המערכות האחרות בארגון וניזונות בנתונים הנגזרים מהמערכות האחרות ומאורגנים במבנה אינטגרטיבי וייעודי ליישומי תמיכה בקבלת החלטות על ידי מערכות אחרות. תפיסת מחסני הנתונים מבוססת על טכנולוגיות מחשוב מודרניות ומהווה מרכיב חשוב בתהליך ההופך נתונים להחלטות עסקיות. בשכבת מחסן הנתונים נמצא טכנולוגיות וכלים לעיבוד אנליטי וניתוח מידע, וביניהם: מחוללי שאילתות ודוחות, מערכות מידע למנהלים (EIS), מערכות ניתוח מידע רב-מימדי (Multi Dimensional Data Analysis), מערכות לכריית נתונים (Data Mining) ועוד. בנושאים אלה דן ספר זה.

המגמות העיקריות בתחומי טכנולוגיית המידע

הבה נסקור בקצרה את המגמות העיקריות בתחום טכנולוגיית המידע. מתוך כלל השינויים והתהפוכות הקשורים לצד הטכנולוגי, נצביע על המגמות האסטרטגיות העיקריות הבאות: השיפורים ביחסי עלות/ביצועים של החומרה, המעבר למחשוב מבוזר מבוסס שרת/לקוח, הופעת האינטרנט והאינטראנט והשימוש ההולך וגדל בטכנולוגיית האובייקטים. נפרט בקצרה כל אחת ממגמות אלו:

❖ **ירידה מתמדת בעלויות החומרה, תוך עלייה מתמדת בביצועיה:** במשך השנים אנו עדים לשיפור בלתי פוסק בביצועי המעבדים, מערכות חומרה מבוססות על ריבוי מעבדים ועיבוד מקבילי, שיפור מתמיד בביצועי הזיכרון המרכזי ובנפחו, הגדלת קיבולת האחסון בדיסקים, מהירות גבוהה יותר להעברת נתונים בין התקני

קלט/פלט לבין הזיכרון הפנימי, ועוד. שיפורים בלתי פוסקים אלה בעוצמת החומרה מלווה בירידת עלויות, דבר שהפך רכישת שרתים בעלי עוצמה גדולה לנחלת רוב הארגונים, ולא רק ארגוני הענק.

❖ **מעבר לסביבת שרת/לקוח ומחשוב מבוזר (Client/Server and Distributed Computing):** ממחצית שנות ה-80 ניתן לזהות מגמה ברורה של מעבר מיישומי מחשוב מבוססי מחשב מרכזי ליישומי מחשוב המבוססים על טכנולוגיות שרת/לקוח ומחשוב מבוזר. בשנים האחרונות אנו עדים להתפתחויות רבות ומרשימות בתחומים אלה: עוצמת המחשוב של המחשב האישי, עוצמות חישוב ומהירויות עיבוד, העוצמה של ממשק המשתמש הגרפי (Graphical User Interfaces), שרתים רבי עוצמה וזולים יחסית הפועלים עם מערכת הפעלה פתוחה כמו Unix ו- Windows NT, בסיסי הנתונים הטבלאיים (Relational Data Bases), רשתות התקשורת המקומיות (Local Area Networks), תוכנות קישור (Middleware) בין פלטפורמות מחשוב הטרוגניות ועוד. התפתחויות אלו תרמו לגל הגואה של הופעת יישומי מחשב מתוחכמים, בעלי פונקציונליות מורכבת, השונים באופן מהותי מהיישומים שהכרנו עד כה.

הארגון המודרני הנדרש להתמודד בסביבה דינמית הולכת וגדלה, מצא שיישומים המבוססים על תפיסת שרת/לקוח מספקים רמה גבוהה יותר של פונקציונליות, גמישות ותגובה. למרות הקשיים שבהם נתקלו הארגונים בתפעול ובתחזוקת יישומי שרת/לקוח, ולמרות האכזבה מכך שמודל מחשובי זה לא הביא להוזלה המובטחת בעלויות, ניתן לומר בוודאות גבוהה שתפיסת המחשוב המבוזר היא התפיסה המועדפת על הארגונים. כל זאת, למרות הקושי לנהל, להפיץ ולתחזק את היישומים בסביבה זו. על רקע זה אנו עדים להתפתחות מודל שרת/לקוח, המבוסס על טכנולוגיות אינטרנט.

❖ **האינטרנט והמסחר האלקטרוני (Internet/Intranet and Electronic Commerce):** מהפיכת האינטרנט התרחשה במהירות עצומה. האינטרנט ענה על צורך אמיתי של שיתוף והפצת מידע לכמויות הולכות וגדלות של משתמשים פנימיים בארגון, ללקוחות הארגון ולשותפיו העסקיים. רשת האינטרנט מייצגת שיטה זולה ויעילה להגיע אל המשתמשים ואל הלקוחות, ולמעשה מהווה את ראשיתו של תהליך החלפת השוק הפיסי והמקומי בשוק מסחר אלקטרוני מדומה (Virtual Marketplace) וגלובלי. בנוסף, מציג מודל המחשוב המבוסס על תפיסת האינטרנט והתקנים שלה, מודל נוח של הפצת ותחזוקת תוכנה בסביבה מבוזרת והטרוגנית, נושא שסביבת שרת/לקוח רגילה מתקשה להתמודד עמו. סביר להניח, שעם הזמן נהיה עדים ליישומים רבים יותר המבוססים על טכנולוגיות אינטרנט, אשר יחליפו בהדרגה את היישומים מבוססי מודל שרת/לקוח המקובל.

❖ **טכנולוגיה מוכוונת אובייקטים (Object Oriented Technology):** הצורך המתמיד בשיפור תהליכי פיתוח התוכנה ובשיפור היצרנות של המפתחים הוביל למגמה הולכת ומתחזקת של שימוש בטכניקות ניתוח, עיצוב ופיתוח תוכנה המבוססות על מודל האובייקטים. טכנולוגיית האובייקטים טומנת בחובה את ההבטחה שניתן יהיה להגיע לתהליכי ייצור תוכנה, אשר מבוססים על רכיבי תוכנה

מוכנים מראש (Component Based Development), בכוונה לקצר משמעותית את תהליכי הפיתוח. השימוש ברכיבי תוכנה מוכנים מראש גם ישפר את אמינות התוכנה המתקבלת ויאפשר בניית מערכות תוכנה מורכבות הבנויות מרכיבי תוכנה פשוטים יחסית, שלעיתים גם יפותחו על ידי יצרני תוכנה שונים. סטנדרטים כמו CORBA או DCOM מיועדים לתמוך בתפיסה של בניית מערכת תוכנה מרכיבים המפותחים בנפרד ומבוססים על שיטה אחידה של שיתוף פעולה וקישוריות (Interoperability). מודל האובייקטים משתלב ותומך היטב במגמת המעבר לסביבות שרת/לקוח מבוזרות.

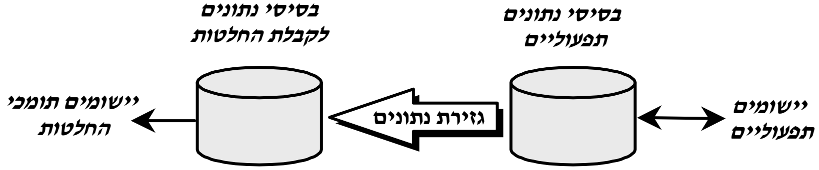
לסיכום, מתוך התבוננות במגמות, הן בצד היישומי והן בצד הטכנולוגי, ומתוך הבנת הארכיטקטורה הכלל ארגונית של מערכות המידע, ניתן לומר בוודאות שתפיסת מחסן הנתונים תהווה את אחת מאבני הבניין החשובות והמרכזיות בטכנולוגיות המידע המודרניות, ומכאן חשיבותו הרבה של הנושא. ארגונים שיבינו את התפיסה וכיצד ליישם ולנצל אותה, ימשיכו להיות תחרותיים ומובילים בתחומם. לא מקרה הוא שמספר כה רב של ארגונים, גם בישראל, מקדישים לנושא את המשאבים ואת תשומת הלב הניהולית הנדרשת לשילוב וליישום התפיסה במסגרת ארכיטקטורת המידע הכוללת של הארגון.

אפיון סוגי היישומים בארגון

כללי

מהדיון עד לנקודה זו עולה שניתן לחלק את מערכות המידע של הארגון לשתי קטגוריות עיקריות: מערכות תפעוליות ומערכות תומכות החלטות. בסעיף זה נעמיק את הדיון ואת האבחנה בין שתי קטגוריות אלו, ונראה שאלו הן שתי קבוצות שונות של יישומים, הדורשות התייחסות שונה ומיוחדת.

הארגון העסקי המודרני מאופיין על ידי ריבוי תהליכי עבודה, שברובם משתמשים במערכות מידע ממוחשבות, שמטרתן לתמוך בתהליכי העבודה השוטפים ולתרום לקיצורם וייעולם. אנו נקרא להם **יישומים עסקיים תפעוליים**. לצד היישומים התפעוליים נמצא אוסף של יישומים העוסקים בהצגת הנתונים ובמצב הארגון. מטרתם העיקרית של יישומים אלה היא סיוע ותמיכה בתהליכי קבלת ההחלטות. ליישומים אלה נקרא **יישומים תומכי החלטות**.



תרשים 1.3: בסיסי נתונים שונים ליישומים שונים.

כדי להבין מדוע, בסופו של דבר, דרושים פתרונות שונים, יש חשיבות רבה להבין את אופיים השונה של שני סוגי יישומים אלה.

יישומים עסקיים תפעוליים (Operational Business Applications)

לכלל היישומים העוסקים בתפעול השוטף של הארגון מקובל לקרוא **היישומים התפעוליים של הארגון** והם הראשונים שכל ארגון מפתח או רוכש ומיישם. יישומים אלה, למרות שלעיתים הם מורכבים ביותר, עוסקים בסופו של דבר במחשוב תהליכי עבודה מובנים וידועים מראש, התומכים בהפעלת הארגון באופן שוטף. המשתמשים העיקריים הם הדרג התפעולי בארגון - עובדים המשרתים את הקהל, מחסנאים, קניינים, עובדים בקו הייצור, אנשי שירות, מנהלי חשבונות, תמחירנים וכדומה. הנתונים הנאגרים בבסיסי הנתונים של יישומים אלה הם בעלי ערך רב, מכיון שבלעדיהם תפעולו השוטף של הארגון בלתי אפשרי. בנוסף לדרג התפעולי משמשים הנתונים במערכות התפעוליות גם את דרג המנהלים, לצרכי קבלת החלטות תפעוליות שוטפות.

המערכות התפעוליות המודרניות מאופיינות בכך שהן **מקוונות** (On Line). כלומר, המשתמשים מבצעים את עבודתם באמצעות מסופים או תחנות עבודה אישיות. עדכון המערכת הממוחשבת מתבצע על ידי **תנועה** (Transaction) שהיא יחידת עבודה לוגית אחת, כגון תנועת כניסה למלאי, תנועת משיכה מחשבון בבנק, תנועת רישום תביעת ביטוח וכדומה. כל תנועה יכולה לעדכן מספר נתונים היכולים להיות מנוהלים בטבלה (קובץ) אחת או יותר. לדוגמה, תנועת רישום כניסת פריט למלאי תגדיל יתרת מלאי של פריט במחסן מסוים, תקטין את הכמות שחייב הספק ותבצע רישום תנועת מלאי חדשה.

ל**תנועה** יש מאפיין של **הכל או כלום** (All-or-Nothing), היא מצליחה או שהיא נכשלת. תנועה אינה יכולה לעדכן רק חלק מהטבלאות ולהיכשל. בדוגמה הקודמת עודכנו שלוש טבלאות שונות במהלכה של תנועה אחת. אם מסיבה כלשהי נכשל עדכון הטבלאות תוך כדי העבודה, תבצע מערכת ניהול בסיס הנתונים פעולת **גלגול לאחור** (Rollback) של התנועה, כדי לבטל את השפעתה על בסיס הנתונים. זו הסיבה לכך שהיישומים התפעוליים נקראים גם **יישומים מקוונים לעיבוד תנועות - OLTP** (On Line Transaction Processing).

יישומים תומכי החלטות (Decision Support Applications)

בנוסף ליישומים התפעוליים, כל ארגון מפתח ורוכש יישומים המציגים את המידע אודות מצב העסק. יישומים אלה אינם עוסקים באופן ישיר בתפעול הארגון, אולם חשיבותם אינה פחותה. בזכותם יכולים מקבלי ההחלטות השונים בארגון לראות את התמונה העסקית ולקבל החלטות באופן שוטף. חשוב להבין שאוכלוסיית המשתמשים של יישומים אלה שונה ובעלת צרכים ודרישות שונים - מנהלים ברמות שונות בארגון, אנשי תכנון ובקרה, כלכלנים שתפקידם לנתח מגמות, מנהלי שיווק.

היישומים לתמיכה בקבלת החלטות עוסקים בניתוח מצב העסק. הם מציגים את מה שקרה, את המשמעויות ומאפשרים לקבל החלטות על מה שצפוי לקרות אם ינקטו פעולות כאלו או אחרות. לא ניתן לומר שהפסקת פעולת יישומים מסוג זה עוצרת את פעילות הארגון, מכיון שהם תומכים יותר בתהליכים ארוכי טווח, ולאו דווקא בפעילות היומיומית השוטפת. להלן מספר דוגמאות ליישומים השייכים לקבוצה זו:

- דיווחים חודשיים לגבי מגמות ברמות המלאי במחסנים.
- ניתוח הצטרפות לקוחות חדשים על ציר הזמן.
- תזרים המזומנים הצפוי.
- ניתוח השפעת שינוי מחיר המוצר על נפח פעילות החברה.
- ניתוח השינויים במכירת המוצרים לפי סוגי המוצר והאזור בארץ.

בשל האופי המאוד שונה של היישומים האלה לעומת היישומים התפעוליים, מקובל לקרוא להם **יישומים תומכי החלטות** (Decision Support Applications) ואת מערכות התוכנה המשרתות אותם מכנים: **מערכות תומכות החלטות - DSS** (Decision Support Systems). יישומים אלה ממוקדים בעיבוד ובניתוח אנליטי של הנתונים והפיכתם למידע בעל משמעות למקבלי ההחלטות השונים בארגון.

השוואת מאפייני שני סוגי היישומים

שתי קטגוריות היישומים שמפעיל הארגון, יישומים תפעוליים ויישומים תומכי החלטות, שונות מאוד זו מזו ובעלות מאפיינים שונים. כדי להדגיש ולהבהיר את ההבדלים ביניהם, נשווה ביניהן על פי מספר קריטריונים. החשיבות בהבנת ההבדלים בין שני סוגי היישומים היא בעיקר בשל המסקנה הבלתי נמנעת: **לא ניתן לספק את דרישות שתי קהיליות המשתמשים באמצעות מערכת מחשוב אחת, ולכן כל סוג יישום צריך לפעול בסביבה שונה ועל בסיס נתונים שונה**. מכיון שהמשמעות של מסקנה זו מרחיקת לכת - שתי סביבות חומרה, שני בסיסי נתונים, כפילות נתונים, אוסף שונה של כלי תוכנה לכל סביבה, עלות גבוהה, תפעול ותחזוקת שתי מערכות מחשוב בארגון - מן הראוי לחקור הבדלים אלה באופן מפורט.

מאפיין	יישומים תפעוליים	יישומים תומכי החלטות
רמת עדכניות הנתונים	***גבוהה ביותר*** נדרשת רמת עדכניות גבוהה. מתבצע עדכון שוטף של בסיס הנתונים, באמצעות תנועות המוזנות קרוב ככל שניתן לרגע היווצרות האירוע.	***נמוכה יחסית*** לא נדרשת רמת עדכניות גבוהה. בדרך כלל דרושים נתונים שמעודכנים לתקופת זמן היסטורית כלשהי - לסוף שבוע קודם, סוף חודש קודם, סוף רבעון וכו'. לדוגמה, כדי לקבל החלטה על מספר עותקים של ספר שרצוי לנהל בספריה, ניתן להסתפק בנתון המעודכן לסוף החודש הקודם המציג את מספר השאלות שבוצעו במהלך 12 החודשים האחרונים ומספר הפעמים שהספר היה חסר.
צורת ארגון הנתונים	***ייעודית עבור תנועות עדכון*** הנתונים מאורגנים בצורה שישרתו בצורה הטובה ביותר את תנועות היישום (קליטת פקודות יומן, רישום עדכון מלאי, פתיחת פקודת עבודה וכדומה) כדי לעמוד באילוצי זמני תגובה קצרים ביותר. מכיון שיישום תפעולי יכול לבצע את אותה תנועה אלפי ומאות אלפי פעמים ביום, חשוב מאוד לארגן את הנתונים בצורה אופטימלית, כדי להשיג את זמני העדכון הקצרים ביותר האפשריים. במשך השנים התפתחו טכניקות עיצוב בסיסי נתונים שעיקר מטרתן לוודא שהנתונים אכן מאורגנים בצורה מתאימה לביצוע התנועות. הטכניקה הידועה ביותר בהקשר זה היא **נירמול נתונים** (Data Normalization). טכניקה זו מביאה לכך שבסיס הנתונים מורכב ממספר רב של טבלאות, שלכל אחת מהן מספר מוגבל יחסית של עמודות.	***ייעודית לשאילתות וניתוחי מידע*** בדרך כלל אין מעדכנים את בסיס הנתונים, אלא רק שולפים ממנו נתונים. בדרך כלל הנתונים מאורגנים לפי נושאים (לקוחות, מוצרים, מכירות וכדומה). מכאן נובע שעקרונות עיצוב, שעיקר מטרתם השגת זמני תגובה קצרים לתנועה, אינם רלוונטיים לסביבת עבודה זו. **נירמול הנתונים** אינו יעד חשוב, ולכן מבצעים לעיתים **דה-נירמול** (Denormalization). היעד החשוב בסביבות אלו הוא נוחות השימוש, פשטות מבנה הנתונים ובהירות, ולכן הנטייה היא לצמצם במספר הטבלאות וליצור מספר רב של עמודות בטבלה.
מספר המשתמשים	***גבוה*** יישומים תפעוליים משרתים כמויות גדולות מאוד של משתמשים, כאשר רובם מבצעים את עבודתם בו-זמנית. נדרש עיצוב מאוד מדויק של התנועות, תוך שימוש בתוכנה מיוחדת - TP Monitor למשל, המאפשרת ניצול יעיל ומתוחכם של כל משאבי המחשב לקבלת זמני תגובה קצרים מאוד.	***נמוך*** משרתים מספר מצומצם יותר של משתמשים. מספרם יכול להגיע למספר עשרות ולפעמים אף פחות.

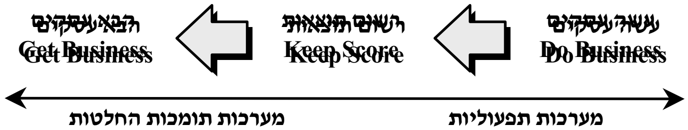
מאפיין	יישומים תפעוליים	יישומים תומכי החלטות
כמות הנתונים המשתתפים באירוע בודד	***נמוכה*** כל תנועה נוגעת בכמות מוגבלת יחסית של נתונים בבסיס הנתונים. כלומר, מספר הטבלאות והשורות המשתתפות בכל תנועה קטן. לדוגמה, תנועה לרישום כניסה למלאי יכולה לעדכן 4 עד 5 טבלאות שונות, ובכל טבלה מספר מוגבל ביותר של שורות.	***גבוהה מאוד*** שאילתות יכולות להיות מורכבות מאוד, ולעיתים דורשות סריקה של כמויות גדולות של שורות וטבלאות. לדוגמה, שאילתה המבקשת להציג שינוי שחל בהיקף המכירות הממוצע לחודש, של מספר מוצרים, לפי משרדי המכירות, בשלוש השנים האחרונות, צריכה לבצע סריקת טבלאות ענקיות, הכוללות את כל המכירות שבוצעו בתקופה הנדרשת.
זמני תגובה	***קצרים מאוד*** נדרשים זמני תגובה קצרים ביותר. מקובל להניח שזמני התגובה לתנועה אופיינית צריכים להיות בתחום של מספר שניות ולעיתים גם חלקי שנייה. זמן תגובה ארוך יותר יכול להביא לדחיית היישום על ידי המשתמשים.	***קצרים יחסית, גמישות גבוהה יותר*** נדרשים זמני תגובה קצרים, אם כי הגמישות רבה. בהחלט ניתן להניח שסוג מסוים של שאילתות יכול להיענות גם תוך מספר דקות, לעיתים אף תוך מספר שעות, ועדיין לא לפגוע ברמת השימוש. עם זאת, סביר להניח שלרוב השאילתות הנפוצות יותר נדרשים זמני תגובה קצרים יחסית.
יכולת תכנון מראש של השימושים	***גבוהה*** משרתים תהליכי עבודה המוגדרים ומעוצבים מראש. מכיון שניתן לתכנן את התנועות מראש, ניתן גם להגדיר מסלולי גישה (אינדקסים), כך שהתנועה תתבצע באופן היעיל ביותר. השימוש בתנועה נעשה בתדירות גבוהה מאוד וניתן אפילו לצפות מראש את מספר התנועות שיישום יבצע (מתוך ידיעת מספר המשתמשים, כמה פעמים בממוצע ליום מבצע כל משתמש תנועה וכו').	***נמוכה*** הגישה ליישומים אלה היא בעלת אופי שימוש מזדמן. בדרך כלל קיים קושי לצפות מראש את סוגי השאילתות ואת מספר הפעמים ששאילתה מסוימת תופעל. מסיבה זו קיים גם קושי בהגדרה מראש של כל מסלולי הגישה שיידרשו.
רמת החשיפה למורכבות מבנה הנתונים	***רמת חשיפה נמוכה*** משתמש הקצה אינו נחשף באופן ישיר למבנה הנתונים ולקשרים ביניהם, מכיון שכל דרישות העיבוד והאחזור מתוכננות מראש. היישומים התפעוליים מפותחים על ידי מפתחים מקצועיים והגישה אל הנתונים מתבצעת רק על ידי היישום. מבנה הנתונים יכול להיות מורכב מאוד ולהכיל מספר גדול של טבלאות. ניווט בין הטבלאות מורכב, אבל היישום מבודד את המשתמש מהנתונים ולכן מורכבות זו חשופה רק למפתחים.	***רמת חשיפה גבוהה*** היישומים מיועדים בדרך כלל לשימושם העצמי של בעלי תפקידים שונים ללא צורך בתיווך אנשי תוכנה ייעודיים. המשתמש הוא זה שמגדיר לעצמו את השאילתות, ולכן גם מבנה הנתונים חשוף בפניו. אם מבנה הנתונים יהיה מורכב מדי הוא יירתע מלנסות ולהבינו, ולכן יש חשיבות רבה בבניית מבנה נתונים פשוט וקל להבנה.

מאפיין	יישומים תפעוליים	יישומים תומכי החלטות
נפח הנתונים	***גדול*** נדרש עומק היסטורי מוגבל (שנה אחרונה, מספר חודשים אחרונים וכדומה). העומק ההיסטורי מוגבל הן בשל כך שהדרישות הן כאלו, וגם בשל הרצון שלא להגיע לנפחי נתונים גדולים מדי, היכולים להשפיע לרעה על זמני העיבוד. לדוגמה, בנקים מנצלים את חלון הזמן של הלילה כדי לבצע עיבודים שונים. ניהול עומק היסטורי גדול מדי בקובץ תנועות של בנק עשוי לגרום לכך שתהליך העיבוד יימשך זמן רב מדי ויגרום לאי עמידה בחלון הזמן הנתון. מסיבה זו משתדלים לנהל את העומק ההיסטורי המינימלי ואת יתר הנתונים להעביר לקבצים היסטוריים מיוחדים.	***גדול מאוד*** נדרש עומק היסטורי רב, כי יישומים אלה מבצעים ניתוחים שונים על ציר הזמן. בסיס הנתונים של מחסן הנתונים משלב נתונים ממספר רב של מקורות, ומכאן נובע גם שנפחי הנתונים המנוהלים ביישומים אלה יכול להיות גדול מאוד. ניהול נפחי נתונים גדולים (VLDB - Very Large DataBases) הוא נושא מורכב הדורש תכנון ומהווה אתגר לא פשוט.
מקור הנתונים	***בעיקר פנים ארגוני*** היישומים התפעוליים מבוססים על הנתונים הנוצרים תוך כדי התפעול השוטף.	***פנים וחוץ ארגוני*** שימוש נרחב בנתונים שמקורם פנים ארגוני, אולם יחד עם זאת נדרשים גם נתונים ממקורות חיצוניים. לדוגמה, אם ארגון מבקש לנתח את נתח השוק שלו, עליו לאסוף גם נתונים חיצוניים אודות מתחריו. נתונים אלה מסופקים בדרך כלל על ידי ארגונים כמו בנק ישראל, הלשכה המרכזית לסטטיסטיקה או ארגונים המתמחים בסקרי שוק ומוכרים מאגרי מידע ייחודיים.
רמת האינטגרציה של הנתונים	***רמה גבוהה*** רמת אינטגרציה גבוהה נדרשת כדי לשמור על אמינות ושלמות הנתונים. המציאות היא שהאינטגרציה אינה מלאה. היישומים התפעוליים מוקמים בדרך כלל על ציר הזמן, לעיתים גם תוך שימוש בפלטפורמות חומרה ותוכנה שונות. ארגונים רבים משתמשים ביישומים שונים הפועלים בסביבות הטרוגניות ומורכבות, כגון יישום אחד הפועל במחשב מרכזי (Mainframe) עם בסיס נתונים IMS או Adabas, יישום חדש יותר הפועל בשרתי Unix עם בסיסי נתונים טבלאיים כמו Oracle או Informix, חבילת תוכנה הפועלת במחשבי מיני	***רמה גבוהה*** נדרש מידע החותך את הארגון באופן רוחבי, ולא לפי היישומים התפעוליים. מבחינת המשתמש הזקוק למידע ניהולי זהו קושי רציני מאוד אם עליו לגשת למיגוון פלטפורמות ובסיסי נתונים שונים, והוא נחשף באופן ישיר למורכבות זו.

מאפיין	יישומים תפעוליים	יישומים תומכי החלטות
	כדוגמת Alpha/VMS עם בסיס נתונים Rdb ויישומים הפועלים בשרתי NT עם בסיס נתונים SQL Server. בדרך כלל נשמרת האינטגרציה על ידי העברות קבצים (File Transfer) בין יישומים, או תוך שימוש בטכניקות קישוריות (Connectivity & Interoperability) מורכבות המבוססות על כלי תוכנה, ולעיתים גם חומרה, מיוחדים (Gateway Servers).	

המסקנה:
שתי סביבות שונות, אבל משולבות

לאחר הסקירה המקיפה של ההבדלים בין שתי קטגוריות היישומים אני מקווה שהקורא אכן השתכנע שמדובר כאן בשתי סביבות מחשוב השונות במטרותיהן ובמאפייניהן, ולכן גם דורשות טיפול ייחודי. בעוד שהמערכות התפעוליות תומכות בעשיית העסקים השוטפת וברישום האירועים השונים במהלכם, עוסקות המערכות תומכות החלטות יותר בצד הבנת מצב הארגון והבאת עסקים חדשים. כמובן שזו הכללה רחבה. ניתן גם למצוא לא מעט מערכות תומכות החלטות שתפקידן הוא שיפור וייעול תהליכי עשיית העסקים היומיומית, תוך ניצול מיטבי של משאבי הארגון. יחד עם זאת, רוב הארגונים שהחליטו על יישום תפיסת מחסן הנתונים התחילו, באופן טבעי, בצד השיווק והמכירות. זאת, מתוך מטרה להבין טוב יותר את לקוחותיהם, את צרכיהם של אותם לקוחות ואת השוק בו הם פועלים.



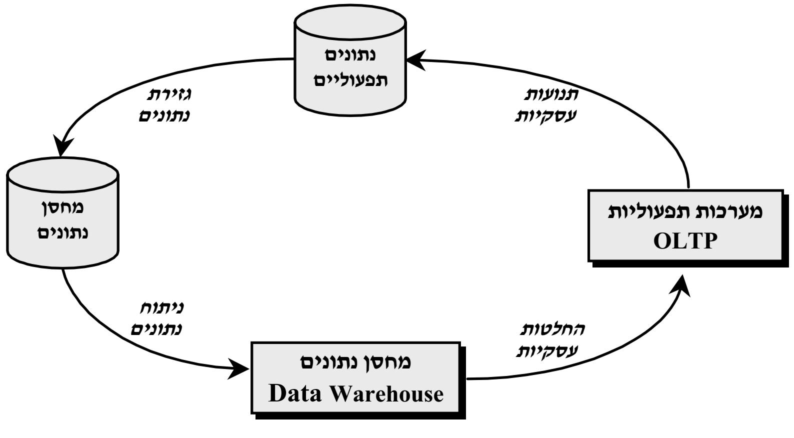
תרשים 1.4: ייעוד שונה למערכות התפעוליות ולמערכות התמיכה בקבלת החלטות.

מחסן הנתונים משמש למטרה שונה מזו של המערכות התפעוליות ותפקידו לענות על שאלות שונות לחלוטין. אם ניקח כדוגמה את המערכת הבנקאית, המערכת התפעולית שלה עונה על שאלות כגון "מה יתרת חשבון הלקוח", "מה היו התנועות שביצע הלקוח בחודש האחרון" או "מה יתרת האשראי ומה ריבית האשראי". אלו שאלות תפעוליות רגילות. לעומתן, מחסן הנתונים צריך להתמודד עם שאלות כגון "לכמה מלקוחות הבנק בעלי יתרה ממוצעת מעל סכום מסוים עדיין אין כרטיסי אשראי", "מהי רווחיות הלקוח על כל המוצרים הבנקאיים שרכש", "לאיזה אחוז מבין לקוחותינו המחזיקים כרטיס אשראי יש גם משכנתה", "מהם מאפייניהם של הלקוחות שנטשו את הבנק בשנה

האחרונה, ולמי מלקוחות הבנק יש לפנות במבצעי שיווק מיוחדים, כדי להקטין את אחוז הנטישה" או "מיהם חמשת הסניפים בעלי הרווחיות הנמוכה ביותר". ברור שזהו אוסף שונה לחלוטין של שאלות בעלות חשיבות עסקית ברמה שונה מהשאלות התפעוליות היומיומיות.

מבחינת הארגון, אלו שתי קבוצות שונות של יישומים שכל אחת מיועדת לפתור בעיה שונה ופועלת בסביבת מחשוב משלה, אך עליהן להיות משולבות ומשלימות. הדרך הנכונה היא להתייחס למערכות אלו כפועלות ב**חוג סגור** (Closed Loop).

- ❖ המערכות התפעוליות מנהלות את בסיס הנתונים התפעולי. כל האירועים השוטפים מוזנים באמצעות יישומים התומכים בביצוע התנועות המעדכנות את בסיס הנתונים.
- ❖ מחסן הנתונים מנהל בסיס נתונים עצמאי המותאם במיוחד לתמיכה בקבלת ההחלטות ולכלי ניתוח הנתונים. מחסן הנתונים מכיל מיגוון כלי ניתוח נתונים שבאמצעותם המשתמשים מנתחים את הנתונים ומקבלים את המידע הדרוש להם לקבלת ההחלטות.
- ❖ להחלטות המתקבלות על סמך המידע המנוהל במחסן הנתונים יש השפעה ישירה, או עקיפה, על המערכות התפעוליות. לדוגמה, לאחר ניתוח אוכלוסיית הלקוחות מתקבלת החלטה להציע לפלח מסוים של לקוחות שירות חדש. שירות חדש זה מחייב שינויים מסוימים במערכות התפעוליות.
- ❖ המערכות התפעוליות קולטות את האירועים הנובעים מהחלטות קודמות. נתונים אלה מועברים למחסן הנתונים לניתוח וקבלת החלטות חדשות, וחוזר חלילה.



תרשים 1.5: מערכות תפעוליות ומחסן נתונים בחוג סגור.

תרשים 1.5 מדגים כיצד פועלות שתי מערכות אלו זו לצד זו ומזינות זו את זו. במציאות, מכיון שלא כל הנתונים מהסביבה התפעולית מועברים אל סביבת מחסן הנתונים, בהחלט ייתכן מצב בו תהליך ניתוח הנתונים מתבסס על נתונים המנוהלים רק במערכות התפעוליות לצד הנתונים המנוהלים במחסן הנתונים.

השלבים בגיבוש תפיסת מחסן הנתונים

כללי

מהיום בו החלו להופיע יישומי המחשב הראשונים עסוק עולם המחשוב בשאלה כיצד לאפשר נגישות לנתונים האגורים ביישומים. ארגונים השקיעו מאמצים ומשאבים רבים בתהליכי רישום האירועים העסקיים השונים (פתיחת הזמנה חדשה, שינוי כתובת העובד, רישום סטודנט לקורס, השאלת ספר לקורא, עדכון מלאי של פריט במחסן, רישום השעות שנדרשו להרכבת מוצר, רישום הפקדה בחשבון, רישום תנועה שבוצעה באמצעות כרטיס אשראי, רישום פרטי שיחה טלפונית שביצע לקוח ועוד), מתוך מטרה ברורה לעשות שימוש בנתונים לשיפור תהליכי עבודה וליכולת קבלת החלטות.

התסכול שהיה מנת חלקם של המשתמשים שרצו את הנתונים במבנה התואם את דרישותיהם הייחודיות ונדחו על ידי יחידות המחשב בגלל הקושי בביצוע המשימה או בגלל העדר המשאבים לביצועה, היה רב. ככל שמספר היישומים בארגון הלך וגדל ופיזור הנתונים בין היישומים הלך וגדל, הלך וגדל גם הקושי בהצגת נתונים ל**רוחב הארגון** (Enterprise Wide). המצב בו נתונים אודות יישות עסקית מסוימת אחת, לקוח למשל, מפוזרים במספר מערכות שונות ופועלות בפלטפורמות מחשוב שונות, הפך למצב רגיל והיקשה מאוד על מקבלי ההחלטות.

השלבים בהתפתחות מערכות לאספקת מידע

לפני שנציג את תפיסת מחסן הנתונים בצורה מפורטת, נסקור בקצרה את התפתחותם של היישומים לאספקת מידע. נעשה זאת מהשלב הראשוני בו נעשה שימוש נרחב בכלים לגישה לנתונים המאוחסנים במערכות התפעוליות, דרך **מרכזי מידע** ועד לתפיסת **מחסן הנתונים**. במילים אחרות, נסקור את השלבים בשחרור הנתונים מ"בית הסוהר לנתונים", כפי שלעיתים נהוג לכנות את המערכות התפעוליות, ועד להבאתם אל מחסן הנתונים; או כמו ששמעתי באחד מימי העיון של חברת **מטה גרופ**: From the Data **Jailhouse** to the Data **Warehouse**.

מחוללי דוחות ושאילתות לגישה לנתונים תפעוליים (Report Writer and Query Generator)

כבר בתחילת שנות ה-70, בעידן היישומים התפעוליים שפעלו בעיקר במחשבים מרכזיים, החלו להופיע כלים שמטרתם היתה לאפשר גישה קלה יותר לנתונים. בשלב זה החל להיווצר הידע הראשוני בפיתוח כלי תוכנה כלליים, שמטרתם לאפשר בנייה מהירה יחסית של דוחות מחשב. השפה העיקרית באותה תקופה לכתיבת יישומי מחשב, שפת COBOL, הורחבה כדי להכיל פקודות מיוחדות לכתיבת דוחות (Report Writer). הרחבה זו נועדה רק עבור המפתחים המקצועיים שכתבו והכירו את השפה.

במשך הזמן החלו להתפתח כלים חיצוניים לשפת COBOL, המיועדים רק לכתיבת דוחות: **מחוללי הדוחות** (Report Generators). כלים אלה היו עדיין מיועדים למפתח המקצועי, אך הם אפשרו פיתוח מהיר יחסית של דוחות. בין הכלים הראשונים שנועדו להחליף ולהשלים את כתיבת הדוחות רק משפת תכנות ניתן לציין את המוצר Easytrieve.

עם הופעת טכנולוגיית בסיסי הנתונים, החלו גם יצרני בסיסי הנתונים לספק כלים ייעודיים שאפשרו כתיבה נוחה של שאילתות ודוחות. שפות שאילתה אלו היו גמישות ואפשרו כתיבה מהירה של שאילתות פשוטות, והכל באופן יחסי.

בעידן זה ענו מחוללי הדוחות והשאילתות על צורך שהלך וגדל - אספקה מהירה של מידע מתוך בסיסי הנתונים התפעוליים. עם זאת, הם לא היו מיועדים לשימוש ישיר על ידי משתמשי הקצה. מטרתם היתה לקצר את הזמן הנדרש על ידי יחידות המחשב להיענות לדרישות המידע של המשתמשים.



תרשים 1.6: יישומים תפעוליים ותמיכה בקבלת החלטות.

ניתן לומר שהחלוצים של מערכות תומכות החלטה היו חברות שניהלו בסיסי נתונים גדולים שהכילו נתוני מכירות ללקוחות, וביקשו לנתח את הנתונים באופן מעמיק יותר ולהבין טוב יותר את לקוחותיהם. חברות אלו החלו להשתמש במחשבים המרכזיים ובמחוללי הדוחות שהיו קיימים באותה תקופה, אולם די מהר התבררה להם האמת המרה: המחשבים המרכזיים הם רבי עוצמה ואמינים מאוד, אך התוכנה שפעלה בהם לא הייתה גמישה במידה מספקת כדי להתמודד עם דרישות קהיליית מקבלי ההחלטות בארגון. די מהר החלו משתמשים אלה להיות מתוסכלים, ובמובן מסוים הפכו יחידות המחשב ל"אויב" שאינו מבין אותם ואינו מספק להם את השירות הנדרש.

סביבות ייעודיות לניתוחים סטטיסטיים במחשבים מרכזיים

במחצית שנות ה-70 החלו להופיע כלים ייעודיים, שהתפתחו מעבודות מחקר באוניברסיטאות ועסקו בעיקר בניתוחים סטטיסטיים של הנתונים. בין הכלים היותר ידועים ניתן למנות את SAS ואת SPSS. מטרת כלים אלה היתה ניתוחים מיוחדים, והם סיפקו טכניקות ניתוח (עיבוד מטריצות, ספריה של פונקציות מתמטיות וסטטיסטיות וכדומה). בשלב זה החלו להופיע גם ניצנים ראשונים של כלים המיועדים לניתוחים פיננסיים, שכללו נושאים כמו סדרות עיתיות (Time Series Analysis), כריית נתונים לעומק (Drill Down), ניתוחי What If, חיפושי מטרה ועוד.

כל הכלים השייכים למשפחה זו היוו התקדמות משמעותית, לעומת הקושי בגישה ובשליפה של נתונים מהמבנים הקשוחים של הקבצים, או בסיסי הנתונים ההירככיים או הרשתיים שהיו נפוצים באותה עת. במובן מסוים, ניתן לראות בכלים אלה דור ראשון ופורצי דרך עבור כלי התוכנה המתוחכמים של ימינו, כלי **כריית הנתונים**.

חלק מכלים אלה, כמו SAS או Focus, המשיכו להתפתח, להרחיב את הפונקציונליות שלהם ולהתאים את עצמם לטכנולוגיות שרת/לקוח מודרניות והם ממשיכים להיות בשימוש נרחב גם כיום.

מערכות תומכות החלטה (DSS - Decision Support Systems)

השלב הבא בהתפתחות מערכות אספקת מידע היה עם הופעת מערכות תוכנה מיוחדות שנקראו מערכות תומכות החלטות. כלים ראשונים אלה היו מבוססים על מחשב מרכזי ועבודה עם מסופים והתמחו ביכולת הצגת נתונים מתוחכמת בחתכי מידע שהוכנו מראש. בשל כך הם דרשו תהליכי הכנת נתונים מורכבים למדי והתבססו על ארגון נתונים ייחודי, כדי לאפשר זמני תגובה מהירים. הם גם הכילו אפשרויות מגוונות לבניית מודלים עסקיים מורכבים, לביצוע חישובים מורכבים ולהצגת הנתונים בצורות מתוחכמות (יחסית לאפשרויות שהיו קיימות באותה עת). מכיון שהכלים התבססו על מבנה נתונים ייחודי ודרשו גזירת נתונים ממערכות תפעוליות, נוצרה לראשונה ההזדמנות לקבלת תמונת נתונים ארגונית, ולאו דווקא קבלת נקודת המבט של היישום הבודד.

הכלים ששימשו למערכות תומכות החלטה היוו את הבסיס למערכות הניתוח הרב ממדיות המודרניות של ימינו, וניתן למנות ביניהן מוצרים של חברות כגון Pilot, Comshare, Express ועוד. התפיסה העומדת מאחורי כלים אלה היא של **קוביה רב ממדית** (Hypercube) המאחסנת את הנתונים בצורה ייחודית, לאחר שעברו תהליכי הכנה מיוחדים.

מערכות מידע למנהלים (EIS - Executive Information Systems)

כלים אלה החלו להתפתח עם הופעת המחשבים האישיים והממשקים הגרפיים ונועדו לקהל מטרה מאוד מסוים: המנהלים הבכירים בחברה. מנהלים אלה תמיד עסוקים והזמן שהם יכולים להקדיש לניתוח הנתונים מועט מאוד. בדרך כלל הם בוחנים את הארגון ברמה סיכומית גבוהה, וגם כשהם מבקשים לרדת לפרטים, הירידה לפרטים מוגבלת למדי. בשל אופיים המיוחד של משתמשים אלה, מבוססות מערכות המידע למנהלים על דוחות ונתונים שהוכנו מראש, במבנה המותאם לצורת ההסתכלות של המנהל הבכיר, ולכן הן התאימו רק לאותם חתכים וניתוחים שהוגדרו מראש. לדוגמה, מערכת המציגה את המכירות של חברה בכל העולם, בכל יבשת, בכל ארץ ובכל סניף. המנהל יכול לנוע בהירככיה זו מעלה או מטה, וזהו. הוא יכול להגדיר ערכי סף מסוימים, ואם הנתונים חורגים מערכי סף אלה הוא יכול לקבל את החריגים בטכניקות שונות: צבע

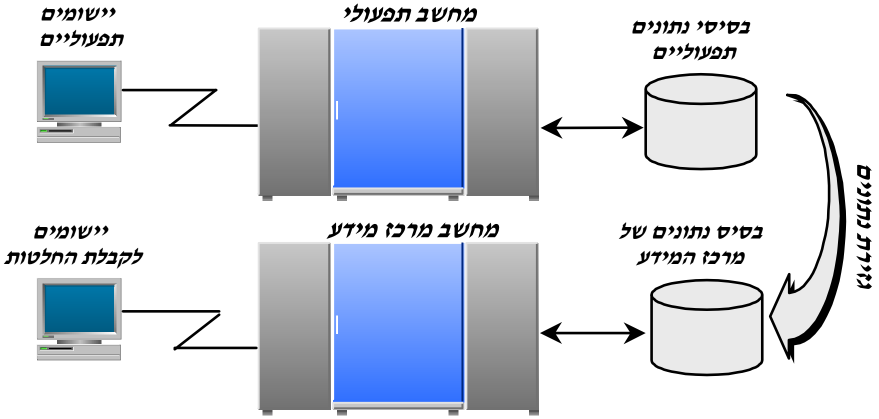
שונה, הדלקת רמזור אדום וכדומה. למשל, אם המכירות בארץ מסוימת ירדו מתחת לסף מסוים, ניתן לבדוק את המכירות בכל סניף, לזהות באיזה סניפים היתה ירידה ולבקש ממנהל המכירות של אותו סניף את ההסבר לכך.

לעומת יכולת ההצגה, בכל הקשור ליכולת הניתוח האנליטית היוו מערכות אלו נסיגה לעומת המערכות לתמיכה בקבלת החלטות. בשל יכולות הניתוח המוגבלות ומחירן הגבוה, לא זכו מערכות אלו להצלחה גדולה. קושי נוסף נבע מכך שמערכות אלו יצאו מנקודת הנחה שהמנהל הבכיר אינו מבין במחשבים ואינו אוהב להשתמש במחשב, ולכן יש צורך לבנות את המערכות באופן הדורש אינטראקציה מינימלית בלבד. הנחות אלו היו אולי נכונות במידה מסוימת לאותה עת, אבל הן בוודאי אינן נכונות לימינו אלה.

מרכזי מידע (Information Centers)

בתחילת שנות ה-80 החל עידן **מרכזי המידע**, תפיסת מחשוב שפותחה וגובשה על ידי חברת **יבמ**. זו היתה הפעם הראשונה שהחלה להתגבש סביבה מחשוב נפרדת וייעודית, שמטרתה שיפור בתהליכי אספקת המידע למשתמשים. התפיסה דיברה על אפשרות לגישה ישירה של המשתמשים אל בסיס נתונים ייעודי, ללא התערבות מתכנתים. בסיס הנתונים הייעודי נבנה על ידי גזירת נתונים מהמערכות התפעוליות של הארגון, והגישה אליו מתבצעת באמצעות כלי תוכנה מיוחדים וקלים לשימוש. מרכזי המידע מומשו במחשבים מרכזיים, פלטפורמת המחשוב שהיתה נפוצה באותה עת.

בגלל המחיר הגבוה של המחשבים המרכזיים וכלי התוכנה המיועדים למרכזי המידע, וגם בשל הצורך באנשי תמיכה לניהול ותפעול מרכז המידע, נעשה מאמץ גדול להגיע לסטנדרטיזציה של הכלים. בסופו של דבר, בגלל המורכבות הגדולה יחסית של הכלים וחוסר גמישותם, נשארה הגישה הישירה של המשתמשים לבסיס הנתונים בגדר חלום עבור רבים מהם, ולמעשה פותחו יישומים תפורים שפעלו בסביבה זו. רובם הפכו לסביבה מונחית תפריטים, בה יש מיגוון רחב יחסית של דוחות ושאילתות מוכנות מראש.



תרשים 1.7: סביבת מרכז המידע.

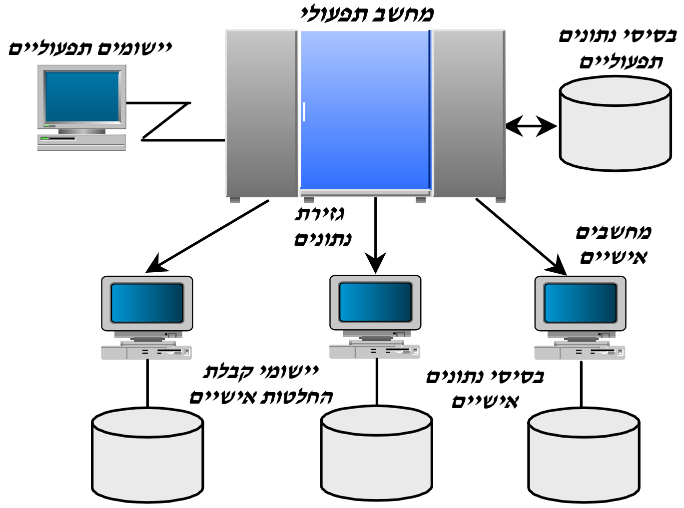
ניתן לומר שמרכזי המידע נחלו הצלחה מוגבלת בלבד. הם לא עמדו בתקווה לשמש מנוף משמעותי להפיכת נתונים למידע על ידי המשתמשים בכוחות עצמם, ובגמישות הנדרשת בסביבה העסקית. מרכזי המידע התבססו על טכנולוגיה יקרה ולא גמישה, ולכן לא שינו באופן מהותי את נוף המחשוב. בסופו של דבר הם היו פתרון נקודתי עבור חלק מהמשתמשים, בעוד שאחרים נמנעו מלנצל תפיסה זו.

עם זאת, חשיבות מרכזי המידע נבעה מהעובדה שהם פרצו את הדרך ושימשו כחלוץ בהבנה שיישומים לאספקת מידע זקוקים לסביבת מחשוב נפרדת וייעודית. פריצת הדרך האמיתית המתינה לחידושים הטכנולוגיים, שהבשילו רק לקראת סוף שנות ה-80 ותחילת שנות ה-90 - טכנולוגיית שרת/לקוח, מחשבים אישיים רבי עוצמה, ממשק משתמש גרפי, בסיסי נתונים טבלאיים וכלי תוכנה לשאילתות ולניתוחים רב-ממדיים. התפתחויות אלו, יחד עם הצורך הגובר לאספקת מידע לתמיכה בתהליכי קבלת ההחלטות, סללו את הדרך לתפיסת המודרנית של **מחסן הנתונים**.

סביבות תומכות החלטות מבוססות מחשבי אישיים

עם הופעת המחשבים האישיים ותפוצתם הנרחבת בארגונים העסקיים החלו המשתמשים העוסקים בקבלת החלטות לבנות לעצמם סביבת עבודה נפרדת ומיוחדת ליישומים אלה. כלי התוכנה העיקריים - הגיליון האלקטרוני ובסיסי הנתונים האישיים - שימשו תחליף נאות לכלים הלא גמישים של המחשב המרכזי. גם פתרון זה לא היה נקי מתקלות: יישומים מורכבים לתמיכה בקבלת החלטות דרשו כלי תוכנה מורכבים יותר מאשר הגיליונות האלקטרוניים, וכמובן שהם היו גם יקרים ומסובכים יותר. הצורך בגזירת הנתונים מהמערכות התפעוליות היה מורכב ומתסכל, וחייב לעיתים הזנה מיוחדת וכפולה של הנתונים. הזנה כפולה זו וגזירות הנתונים הייחודיות לכל משתמש גרמו לחוסר תאום ולחוסר עקביות בנתונים, ולוויכוחים בין משתמשים ומחלקות שונות לגבי נכונות הנתונים. אנשי המחשוב בארגון עמדו מתוסכלים מול הגל הגואה של בקשות לגזירת נתונים בחתכים ובמבנים מיוחדים. אותם נתונים נגזרו פעם אחר פעם, לשימוש מחלקות שונות ובגלל ההגדרות השונות בתהליכי הגזירה גם גרמו לחוסר עקביות ובלבול המשתמשים.

בתוך פרק זמן לא ארוך הגיעו הארגונים למסקנה שיישומי תמיכה בקבלת החלטות, המבוססים על מחשבים אישיים ובסיסי נתונים מקומיים מנותקים, אינם מהווים פתרון הולם לקבלת מידע תומך החלטות. הפתרון הנכון היה, כמובן, להתבסס על בסיס נתונים ייעודי ומרכזי. בסיס נתונים זה מנוהל על ידי פונקציית המחשוב בארגון, תוך שילוב המחשבים האישיים וכלי התוכנה המתוחכמים שלו. זו, כמובן, גישת מחסן הנתונים המודרני, המבוססת על טכנולוגיית **שרת/לקוח**.



תרשים 1.8: יישומי קבלת החלטות אישיים.

מחסן הנתונים (Data Warehouse)

הניסיונות לבנות סביבות ייעודיות לאספקת מידע לתהליכי קבלת החלטות נמשכו שנים רבות, אולם ללא הצלחה יתרה. חוסר ההצלחה נבע מכך שהטכנולוגיה לא אפשרה מענה הולם לדרישות קהיליית המנהלים ומקבלי ההחלטות. משתמשים אלה זקוקים לממשקים גרפיים נוחים, כלי ניתוח מתוחכמים, כלים סטטיסטיים וטכניקות מיוחדות להצגת הנתונים (Visualization). לשם כך דרושים עוצמת עיבוד חזקה במיוחד וממשקים גרפיים מתוחכמים, ולכן רק טכנולוגיית שרת/לקוח מהווה פתרון הולם לדרישות אלו. הבסיס לתפיסת מחסן הנתונים הוא היכולת לנהל את הנתונים בשרת בסיס נתונים רב-עוצמה ולשלב אותו עם תחנות עבודה בעלות ממשקים גרפיים מתוחכמים.

מבחינת הארכיטקטורה הכללית דומה מחסן הנתונים למרכז המידע. הוא מבוסס על גזירת הנתונים מהמערכות התפעוליות אל מערכת ייעודית לתמיכה בקבלת החלטות. מבחינת הטכנולוגיות המשמשות את מחסן הנתונים ואשר שימשו את מרכזי המידע, יש הבדל מאוד מהותי. מחסן הנתונים מבוסס על טכנולוגיות מידע מודרניות ומשתמש במיטב החידושים שהחומרה, התקשורת והתוכנה מאפשרים כיום. המעבר ממרכזי המידע לתפיסת מחסן הנתונים היא הרבה יותר מאשר שינוי במונחים. מחסן הנתונים מבוסס על טכנולוגיה מחשוב מתקדמת, מיועד לקהל רחב יותר ולסוגי משתמשים רבים ומגוונים ומבוסס על ארכיטקטורה כוללת ועל מתודולוגיה ברורה ליישום.

הערך העסקי של מערכת מחסן נתונים טמון במידה רבה במילה **גילוי** (Discovery). שילוב הנתונים ממספר מערכות תפעוליות ואחסונם במאגר מרכזי אחד מאפשר למשתמשי הקצה גישה נוחה לנתונים ב**חתך נושאי** (Subject Oriented) לרוחב הארגון. כלים של משתמשי הקצה מאפשרים התבוננות רב-ממדית, כריית נתונים, ניתוח סטטיסטי ומאפשרים להם לשאול שאלות, לחקור את הנתונים ולגלות עובדות חדשות אודות הארגון ופעילותו העסקית.

מחסן הנתונים שובר את המחיצות בין איי הנתונים ומשחרר אותם לשימושים של ניתוח ותמיכה בקבלת החלטות. הקמת מחסן הנתונים מהווה הזדמנות לארגון לעצב מחדש (Reengineer) את תהליכי העבודה שלו, ללא צורך בשינוי מהפכני במערכות התפעוליות. לעיתים, המידע המתגלה תוך כדי עבודה עם מחסן הנתונים, מאפשר לארגון לשנות תהליכים עסקיים מרכזיים, כדי להפוך לארגון תחרותי יותר, יעיל יותר ובסופו של דבר - רווחי יותר. ההבטחה שניתנה בעבר על ידי מערכות תומכות ההחלטה (DSS) מתממשת, סוף כל סוף, בדמות תפיסת מחסן הנתונים.

סיכום

בפרק זה סקרנו את השוני בין מערכות מידע תפעוליות לבין מערכות מידע שמטרתן תמיכה בקבלת החלטות. הצגנו את ההתפתחות של מערכות אלו - ממערכות שפעלו במחשבים מרכזיים ועד למערכות מחסן נתונים המבוססות על טכנולוגיות מחשוב מודרניות. כיום, קיימת הסכמה רחבה שתפיסת מחסן הנתונים תתפוס תאוצה רבה ואת תשומת ליבם של הארגונים. למערכות אלו חשיבות מרובה לצד המערכות התפעוליות. מחסן הנתונים מהווה נדבך חשוב ביכולתן של מערכות תפעוליות לשרוד ולהמשיך ולשרת את הארגון. מערכות אלו מאפשרות לארגון להמשיך ולהפעיל מספר רב של יישומים תפעוליים הפועלים בפלטפורמות שונות ועם כפילות נתונים ביניהן. הדבר מתאפשר מפני שמחסן הנתונים מרכז את הנתונים מכל המערכות, פותר חלק מהסתירות הקיימות בין המערכות התפעוליות השונות ומספק לארגון נקודת מבט כוללת ואינטגרטיבית - Single Image of Business Reality.

אחת הטעויות הנפוצות היא שתפיסת מחסן הנתונים מתאימה רק לארגונים המתבססים על מחשבים מרכזיים עבור המערכות התפעוליות. האמת היא שאין כל קשר בין תפיסת מחסן הנתונים לבין הפלטפורמה בה משתמש הארגון למערכות התפעוליות שלו. תפיסת מחסן הנתונים מתאימה באותה מידה גם לארגונים המתבססים על יישומים תפעוליים מודרניים הפועלים בסביבות שרת/לקוח ואינטרנט. כל הבעיות שמנינו במערכות התפעוליות אינן נובעות מהטכנולוגיה, אלא מעצם מהותן של מערכות תפעוליות המיועדות לתמיכה בעשיית העסקים וברישום התוצאות, ואינן מיועדות לשרת יישומים תומכי החלטות.

פרק 2:
ארכיטקטורת מחסן הנתונים

מבוא

בפרק זה נגדיר את תפיסת מחסן הנתונים, נציג את הארכיטקטורה הכוללת שלו ונפרט את מכלול הטכנולוגיות והכלים המרכיבים אותו.

תפיסת מחסן הנתונים מספקת מסגרת כוללת לניהול נתונים המיועדים לתמיכה בקבלת החלטות ומתן דגש על היעילות, העקביות והנוחות. מחסן הנתונים מבוסס על בסיס נתונים ייעודי ונפרד, המכיל נתונים אודות פעילות הארגון וסביבתו. הוא משמש באופן בלעדי את תהליכי ניתוח הנתונים ותמיכה בקבלת ההחלטות. בסיס נתונים ייעודי זה מכיל נתונים הנשלפים ממקורות שונים, תוך ביצוע תהליכי גזירה, שיפור ושילוב מיוחדים. מטרתו אחת ויחידה - לספק את צרכי המידע של קהיליית משתמשים חשובה - **מקבלי ההחלטות בארגון**. בסיס נתונים זה חייב להיות זמין, עדכני, אמין ומאורגן באופן התואם את היישויות העסקיות העיקריות ואת תהליכי קבלת ההחלטות. חשוב להבין שבסיס נתונים, שמטרתו לתמוך בתהליכי קבלת החלטות מורכבות ודינמיות, חייב להיות מעוצב ומאורגן באופן מיוחד, התואם את מטרותיו. ארגון זה שונה באופן מהותי מהדרך בה מאורגנים הנתונים המשמשים ומשרתים את המערכות התפעוליות.

מן הראוי לציין שתחום מחסני נתונים הוא תחום צעיר יחסית, ולכן הוא עדיין בתהליך התגבשות. המושגים עדיין אינם מוגדרים עד הסוף, הטכנולוגיה ממשיכה להשתנות, הארגונים עדיין נמצאים בתהליך יישום והפקת לקחים ולכן גם מתודולוגיות היישום הן עדיין בחיתוליהן. מכיון שתחום זה מתפתח בקצב מהיר, יש לצפות לכך שדברים ישתנו, הגדרות מדויקות יותר תתגבשנה וטכנולוגיות חדשות תופענה ותחלפנה טכנולוגיות קיימות. אם זו חשיפתו הראשונה של הקורא לתחום מחסן הנתונים, אני מקווה שדינמיות וחוסר ההתגבשות הסופית של התפיסה לא תרפה את ידיו. זהו אך תהליך טבעי בהתפתחותה של תפיסה הצומחת ותופסת פופולריות עצומה.

מהו מחסן נתונים?

ברמה הבסיסית ביותר, **מחסן נתונים** היא תפיסה כוללת העוסקת באיסוף, אחסון, הפצה ושיתוף של נתונים למטרת תמיכה בקבלת החלטות. ניתן לומר שזו תפיסת קצה לקצה המטפלת בכל מיגוון התהליכים והפעילויות הדרושות כדי לספק את הנתונים הדרושים לתמיכה בקבלת החלטות עסקיות. מחסן הנתונים שואב את הפונקציונליות שלו מתוך העובדה שהנתונים נגזרים ממקורות שונים, עוברים תהליכי שיפור וטיוב ומשולבים יחדיו. מחסן הנתונים שואב את עוצמתו מהקלות בה מוצגים נתונים אלה ומנותחים על ידי המשתמשים בו.

אחת ההגדרות המצוטטת ביותר שייכת למי שנחשב לאבי התפיסה, Bill Inmon, שבמאמר משנת 1992 הציג את ההגדרה הבאה:

> **מחסן הנתונים (Data Warehouse)** מוגדר כאוסף נתונים ייעודים ומשולבים, מאורגנים לפי נושא, בעלי עומק היסטורי ושאינם מתעדכנים - אשר מיועדים לתמוך בתהליכי קבלת החלטות.

נתעכב בקצרה על כל אחד ממרכיבי ההגדרה הזו:

❖ **אוסף נתונים ייעודי (Dedicated Data Base):** בסיס הנתונים המשמש את מחסן הנתונים נפרד מבסיסי הנתונים התפעוליים של הארגון. החשיבות בהפרדה זו כבר נדונה בפרק הראשון בספר זה, בו הוסברה האבחנה בין יישומים תפעוליים לבין יישומים לאספקת מידע. העובדה שזהו בסיס נתונים נפרד וייעודי מאפשרת לארגן ולעצב אותו במבנה מיוחד ותואם למטרות נגישות ישירה של המשתמשים. כלומר, שימת דגש מיוחדת על מבנה קל ונוח להבנה מצד אחד, ויעיל - לצורך קבלת זמני תגובה טובים, מצד שני. בסיס נתונים נפרד זה יפעל בשרת ייעודי כדי שלא להעמיס את המערכות התפעוליות, וכדי לאפשר זמני תגובה טובים למשתמשי מחסן הנתונים.

❖ **אוסף נתונים משולב (Integrated Data):** הנתונים של מחסן הנתונים נמצאים **בבסיס נתונים משולב** שבו כל הנתונים הדרושים ליישומים השונים המספקים מידע, ללא תלות במקורם. סביר להניח שבארגון יהיו מספר בסיסי נתונים תפעוליים שונים הפועלים על פלטפורמות חומרה ותוכנה שונות. על כן, היכולת לשלב את כולם לסביבה משולבת אחת היא בעלת חשיבות רבה. שילוב זה יבטיח אחידות ונוחות מנקודת מבט המשתמש.

❖ **ארגון לפי נושא (Subject Oriented):** הנתונים במחסן הנתונים מאורגנים לפי נושא, ולא לפי היישום המייצר אותם. בדרך כלל, הנתונים במערכות התפעוליות מאורגנים לפי צרכי היישום, לפי דרישות הביצועים ולפי אופי התנועות של היישום. סביר להניח שנתונים הקשורים ליישות עסקית אחת, למשל לקוח, יהיו מפוזרים במספר יישומים תפעוליים שונים, שכל אחד מהם פותח בזמן שונה ולעיתים גם על פלטפורמה שונה. המשתמש הטיפוסי של מחסן הנתונים זקוק לנתונים אודות היישויות העסקיות לרוחב הארגון, כמו למשל כל הנתונים על לקוח מסוים.

הדוגמה הנפוצה ביותר היא נתוני לקוח במערכת בנקאית. אלה מפוזרים ביישומים שונים (עובר ושב, חסכונות, השקעות, משכנתאות וכדומה). דרישה בסיסית של משתמש במחסן הנתונים היא להיות מסוגל לנתח את נתוני הלקוח השונים ואת פעילותו בבנק. כפי שנראה בהמשך, ארגון הנתונים לפי נושא מציב אתגר מיוחד, מכיון שהוא דורש עיצוב מיוחד וגזירת הנתונים ושילובם לקראת הכנסתם למחסן הנתונים.

- **בעל עומק היסטורי (Time Variant):** מחסן הנתונים מאופיין בעומק היסטורי נרחב יותר מאשר המערכות התפעוליות. חלק ניכר מהניתוחים המתבצעים דורשים ניתוח מגמות, ולכן יש חשיבות מרובה לעומק ההיסטורי. ככל שנעמיק בהיסטוריה, בסיס הנתונים של מחסן הנתונים ילך ויגדל וידרוש משאבי אחסון וניהול גדולים יותר. חשוב להבין שגם במחסן הנתונים רצוי להעריך מהו העומק ההיסטורי הנדרש, כדי שלא לסבך יתר על המידה את תהליכי הניהול ולהאריך את זמני העיבוד ומשאבי המחשב הנדרשים.
- **לא מתעדכן (Nonvolatile):** להבדיל מבסיס הנתונים התפעולי (שצריך לשקף בכל זמן את התמונה העדכנית ביותר, ולכן הוא מתעדכן על ידי תנועות באופן שוטף), מחסן הנתונים אינו מתעדכן על ידי תנועות. הדרך המקובלת לעדכון מחסן הנתונים היא על ידי עדכון במנות תקופתיות (יומיות, שבועיות, חודשיות וכדומה). עדכון מנות זה מתבצע בדרך של **טעינה מסיבית** (Bulk Load), תוך שימוש בתוכניות שירות של בסיס הנתונים, או על ידי הפעלת תוכניות עדכון מיוחדות. רצוי לא לאפשר עדכון ישיר של הנתונים במחסן הנתונים באמצעות תנועות, למעט במקרים קיצוניים מיוחדים.
- **תמיכה בקבלת החלטות (Decision Support):** תמיכה בקבלת החלטות היא מטרת העל של מחסן הנתונים. המאמץ הכרוך בבניית מחסן הנתונים, שיקולי העיצוב, כלי התוכנה המשמשים אותו והארגון התומך והמזין אותו - כולם נועדו למימוש הגישה הקלה והנוחה של משתמשים לסיוע בתהליכי קבלת ההחלטות. מחסן הנתונים מבוסס על אוסף כלי תוכנה מתוחכמים המיועדים להצגה ולניתוח נתונים. כלים אלה מאפשרים גישה ישירה לבסיס הנתונים או בנייה של יישומים מותאמים לתמיכה בתהליכי קבלת ההחלטות.

למרות שהגדרה זו ממצה למדי, חשוב שנדגיש גם מה חסר בה: **המימד הדינמי**. ההגדרה ממוקדת בעיקר בהיבט הסטטי של מחסן הנתונים ומתעלמת מההיבט הדינמי, שהוא אוסף התהליכים והפעילויות הקשורים בפיתוח, תפעול, ניהול, תמיכה ותחזוקת מחסן הנתונים. על רקע זה מצאתי לנכון להביא הגדרה נוספת למחסן הנתונים, המתייחסת אל מחסן הנתונים כאל תהליך, ולא כאל מקום:

> **מחסן הנתונים (Data Warehouse)** הוא תהליך **קצה לקצה** (End-to-End) המארגן נתונים שמקורם במספר יישומים תפעוליים שונים, בהתאם למימד הזמן ובהתאם לנושאים בעלי משמעות למקבלי ההחלטות. בגלל ההיבט התהליכי מקובל גם לקרוא לתהליך זה בשם **מיחסון נתונים (Data Warehousing)**.

כפי שנראה בהמשך, חלק ניכר מהאנרגיה שמשקיע הארגון בהקמת ובתפעול מחסן הנתונים מושקעת בעבודה היומיומית להכנסת הנתונים למחסן הנתונים ובתפעולו השוטף. לכן, מן הראוי שהגדרת מחסן הנתונים תדגיש גם את ההיבט התהליכי, ולא רק את ההיבט הסטטי.

בפרק הקודם סקרנו את ההבדלים העיקריים בין המערכות התפעוליות לבין מחסן הנתונים. להבדיל ממערכות תפעוליות, שהן בדרך כלל בעלות רמת יציבות גבוהה יחסית, מחסן הנתונים מתפתח כל הזמן ומתאים את עצמו באופן שוטף לדרישות המידע המשתנות של הארגון. משמעות הדבר היא שעיצוב מחסן הנתונים חייב לקחת בחשבון את האופי הדינמי - הדרישות של היום בוודאי שלא תהיינה הדרישות של מחר. הבעיה היא שלא ניתן אפילו לחזות מה תהיינה הדרישות העתידיות. ובכל זאת, כפי שנראה בהמשך, ניתן לעצב את מחסן הנתונים כך שתהיה בו מידה רבה של יכולת הסתגלות לשינויים ולדרישות בלתי צפויות. עובדה זו באה לעיתים כהלם למעצבי מערכות, שכל חינוכם בא מתחום המערכות התפעוליות: הם למדו שיש להקדיש זמן ניכר לניתוח הבעיה לפני בניית המערכת, מכיון ששינוי מערכת תפעולית הוא תהליך מורכב ויקר. כאשר הם עוסקים בהקמת מחסני נתונים, הם מנסים להשקיע זמן רב מדי בניתוח ובניסיון לחיזוי הצרכים העתידיים ולעיתים - עד כדי שיתוק (Paralysis by Analysis). לבסוף, כשהם לומדים ומבינים את הדרישות של היום ומנסים ליישם אותן, הדרישות משתנות ולכן מתפתחת אצלם תחושה של מרדף בלתי פוסק אחר דבר שאין להשיגו. כפי שנראה בהמשך, חלק מהתשובה לתכונה זו של מחסני הנתונים טמונה במתודולוגיית הבנייה, העיצוב והיישום.

ארכיטקטורת מחסן הנתונים (Data Warehouse Architecture)

כל מי שנחשף לנושא מחסני נתונים יוצא מבולבל מהאוסף הגדול של השמות השונים: מחסן נתונים ארגוני, מחסן נתונים מחלקתי, מרכול נתונים, מאגר נתונים תפעולי וכדומה. ננסה לעשות סדר באוסף מושגים זה, ונראה שההבדלים ביניהם נובעים מהעושר הרב של ארכיטקטורות ושל היעדים של מחסני הנתונים.

אם נתבונן במחסני הנתונים שהוקמו על ידי הארגונים השונים, סביר להניח שלא נמצא שניים זהים לחלוטין. הסיבה לכך היא שכל מחסן נתונים מותאם לסביבת המחשוב המיוחדת של הארגון, לסביבת החומרה ולכלי התוכנה בהם בחר הארגון להשתמש, ולמיגוון הייחודי של היישומים התפעוליים הפועלים בו. עם זאת, ניתן לומר שקיימים מספר קווי דמיון בולטים בין רוב מחסני הנתונים ורובם כוללים, בצורה זו או אחרת, מספר מרכיבים בסיסי.

מקובל להתייחס למספר ארכיטקטורות עקרוניות של מחסני נתונים, שההבדלים ביניהם נובעים מיעדיהם ומקהל המשתמשים שהם צריכים לשרת:

- ❖ מחסן נתונים ארגוני (Enterprise Data Warehouse)
- ❖ מרכול נתונים (Data Mart)
- ❖ מחסן נתונים רב-שכבתי (Multi Tiered Data Warehouse)
- ❖ מאגר נתונים תפעולי (Operational Data Store)
- ❖ מחסן נתונים מדומה (Virtual Data Warehouse)

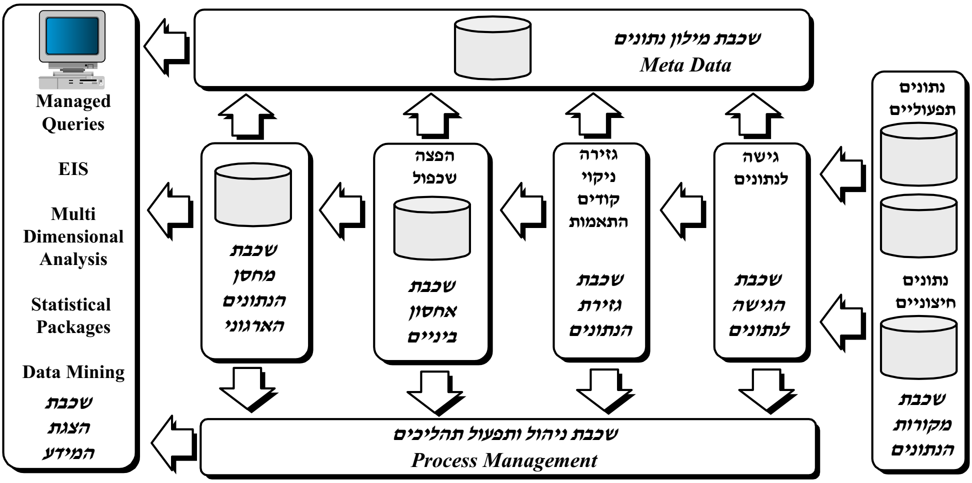
נסקור כל אחת מארכיטקטורות אלו בפירוט.

ארכיטקטורת מחסן נתונים ארגוני (Enterprise Data Warehouse Architecture)

ארכיטקטורת מחסן הנתונים ארגוני היא המקיפה ביותר מבין ארכיטקטורות מחסני הנתונים.

מחסן הנתונים הארגוני הוא מחסן נתונים מרכזי המיועד לשרת את כל המשתמשים בארגון, ולכן הוא משלב בבסיס נתונים אחד את כל הנתונים הנדרשים עבור כל יישומי התמיכה בקבלת החלטות.

באופן טבעי, מחסן נתונים כזה הוא גדול, מקיף ומשרת מספר רב של משתמשים, ולכן תהליך הקמתו מורכב וארוך יחסית. בדרך כלל, פיתוח מחסן נתונים ארגוני מתבצע בשלבים ולא בבת אחת, ובסופו של תהליך ההקמה והיישום יעמוד לרשות הארגון מחסן נתונים מרכזי אחד.



תרשים 2.1: ארכיטקטורת מחסן נתונים ארגוני.

כפי שניתן לראות בתרשים 2.1, הארכיטקטורה הכללית של מחסן הנתונים הארגוני מורכבת ממספר שכבות. נציג אותן כעת ובהמשך נקדיש לחלק מהן הסבר מעמיק יותר.

- **שכבת מקורות הנתונים (Data Source Layer):** את שכבת מקורות הנתונים ניתן לחלק לשתי קטגוריות עיקריות - מקורות נתונים פנימיים ומקורות נתונים חיצוניים. המקורות הפנימיים הם אוסף כל הנתונים שמנהל הארגון במסגרת היישומים התפעוליים; המקורות החיצוניים הם אוסף הנתונים הנדרשים לתהליכי קבלת החלטות ונאספים מחוץ לארגון. הנתונים החיצוניים עשויים לכלול סטטיסטיקות המתפרסמות על ידי גופים רשמיים (כגון הלשכה המרכזית לסטטיסטיקה או בנק ישראל), מאגרי מידע של חברות המנתחות שווקים שונים (כגון Dun&Breadstreet), נתונים דמוגרפיים ועוד. מכיון שאלה הם נתונים חיצוניים לארגון, הם אינם מנוהלים במסגרת המערכות התפעוליות שלו. מקורות הנתונים יכולים להיות מגוונים מאוד. למרות העושר של המקורות החיצוניים, מקור הנתונים העיקרי למחסן הנתונים, אם כי לא הבלעדי, הוא הקבצים ובסיסי הנתונים של מערכות המידע התפעוליות. הבעיה העיקרית היא שמערכות אלו עשויות להיות ישנות מאוד, ולהימצא על פלטפורמות מחשוב מסוגים שונים.

- **שכבת הגישה לנתונים (Data Access Layer):** שכבה זו עוסקת בגישה למקורות הנתונים, הן במערכות התפעוליות והן לנתונים החיצוניים. מכיון שמקורות הנתונים יכולים להיות מגוונים מאוד, קיים לעיתים קושי לגשת אליהם לצרכי גזירת הנתונים. בשנים האחרונות התפתחו כלי גישה שונים לנתונים, כגון EDA/SQL של חברת IBI, מוצרים מבוססי תקן ODBC וכדומה. מוצרים אלה מאפשרים גישה שקופה למיגוון המקורות המבוססים על מבני נתונים שונים (קבצי VSAM; בסיסי נתונים במחשבים מרכזיים כגון IMS או IDMS; בסיסי נתונים טבלאיים כגון Oracle, Informix, Sybase, SQL Server ואחרים). מעבר לתמיכתם במבני נתונים שונים, תומכים כלים אלה בפרוטוקולי תקשורת רבים, במערכות הפעלה ובחומרות שונות.

- **שכבת גזירה, טיוב והמרה של הנתונים (Data Transformation and Enhancement Layer):** שכבה זו עוסקת בתהליך הגזירה וההכנה של הנתונים שיש לאחסן במחסן הנתונים. מכיון שהנתונים המאוחסנים בבסיסי הנתונים התפעוליים אינם מתאימים למטרות תמיכה בקבלת החלטות, יש לבצע מספר פעולות מקדימות. בדרך כלל, בסיום שלב זה נמצאים הנתונים במבנה מתאים לטעינה למחסן הנתונים.

- **שכבת אחסון ביניים והעברה (Staging Data Layer):** שכבה זו מאחסנת לתקופת ביניים את הנתונים המוכנים לטעינה. לעיתים נוח לטפל בנתונים מראש ולצבור אותם לפני טעינתם. סיבה עיקרית לכך היא נפחי הנתונים ו/או ניצול חלון זמן מתאים. הנתונים נשמרים בשכבת ביניים זו עד לטעינתם למחסן הנתונים והיא גם יכולה להפיץ אותם לאחר מכן. משכבה זו מועברים הנתונים אל מחסן הנתונים, לשם טעינתם לבסיס הנתונים. שכבת פעילות זו אינה דרושה בכל מימוש של מחסן נתונים.

- **שכבת בסיס הנתונים (Data Base Layer):** שכבה זו מנהלת את בסיס הנתונים המרכזי של מחסן הנתונים. רוב יישומי מחסן נתונים משתמשים בבסיס נתונים טבלאי. הנתונים נטענים אל בסיס הנתונים באמצעות תוכניות שירות או יישומים מיוחדים המעדכנים אותו. בדרך כלל, עם תום שלב הטעינה נבנים האינדקסים, מחושבים סיכומים שונים ומבוצעות פעולות ניהול, כנדרש.

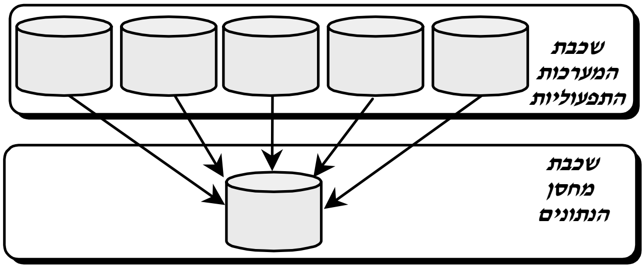
- **שכבת הצגת המידע (Information Presentation Layer):** שכבה זו עוסקת בהצגת המידע המנוהל במחסן הנתונים, וכוללת אוסף של כלי תוכנה ויישומים המיועדים למשתמשי מחסן הנתונים. בדרך כלל, אין בנמצא כלי אחד ויחיד המתאים לצרכי כל המשתמשים. על כן, מקובל לראות בשכבה זו מספר כלים שונים המיועדים לצרכים שונים: מחוללי שאילתות ודוחות, כלים לניתוח רב-מימדי, כלים לניתוחים סטטיסטיים, כלים לבניית מודלים פיננסיים, גיליונות אלקטרוניים, כלי כריית נתונים (Data Mining) וכדומה. בחלק מהמקרים המשתמשים עובדים באופן ישיר עם כלים אלה ובמקרים אחרים נבנים עבורם יישומים ייחודיים, הכל בהתאם ליכולות המשתמש וצרכי המידע שלו. קיימים מצבים בהם המשתמשים זקוקים לאותו מבנה מידע באופן תקופתי. לכן, מקובל להריץ אוסף דוחות מול מחסן הנתונים ולהעמיד את התוצאות לרשות אותם משתמשים. כך, המשתמשים אינם צריכים לנסח את המבנה מחדש בכל פעם ואינם צריכים לזכור כיצד להפעילו. טכנולוגיית האינטרנט הפכה לטכנולוגיה נפוצה מאוד לצורך הפצת המידע המופק ממחסן הנתונים למשתמשים השונים.

- **שכבת קטלוג הנתונים (Meta Data Layer):** שכבה זו משמשת כקטלוג מרכזי המאפשר למשתמש להתמצא בנבכי מחסן הנתונים, במשמעות הנתונים ובעדכניותם. קטלוג הנתונים מכיל את ההסבר לכל הטבלאות, את ההסבר לעמודות שבכל טבלה, כיצד חושבו הנתונים, מה המקור ממנו נגזרו או חושבו הנתונים, למתי הם מעודכנים וכדומה. ללא שכבה זו יש למשתמשים קושי ניכר בהתמצאות במחסן הנתונים ובהבנת משמעות הנתונים האגורים בו.

- **שכבת ניהול ותפעול תהליכים (Process Management Layer):** שכבה זו עוסקת בהפעלת התהליכים השונים הפועלים בסביבת מחסן הנתונים, בתזמון שלהם, בבדיקה שהסתיימו בהצלחה וכדומה.

מכיון שמבחינת ניהול הנתונים בארגון קיימת אבחנה ברורה בין שתי שכבות נתונים - נתונים תפעוליים ונתונים לתמיכה בקבלת החלטות - מקובל גם לקרוא לארכיטקטורה זו, מבחינת הנתונים, **ארכיטקטורה דו-שכבתית** (Two Tier Architecture).

היתרון הגדול בגישת מחסן הנתונים הארגוני הוא, כמובן, מעצם היותו מאגר מרכזי המנהל נתונים אודות כלל היישויות העסקיות המעניינות את הארגון והמשמש את כלל המשתמשים. מכיון שזהו בסיס נתונים מרכזי, ניתן לקבל ממנו חתכי מידע רוחביים חשובים, מעניינים ומגוונים.



תרשים 2.2: שכבות בארכיטקטורת מחסן נתונים ארגוני.

הבעיה העיקרית בגישת מחסן הנתונים הארגוני הוא הזמן הארוך הדרוש להקמתו, תהליך שיכול להימשך לפעמים מספר שנים. הניסיון להקים מחסן נתונים ארגוני דורש להגיע להסכמה בין מספר רב של משתמשים ומחלקות בארגון באשר להגדרת ומשמעות הנתונים, ובאשר לעדיפויות הנושאים השונים שיש לנהל בו. הארגון צריך להשקיע מאמץ וזמן רב בשלב העיצוב והבניה של מודל הנתונים. לעיתים מגיעים לסתירות בין הדרכים השונות בהן משתמשים שונים רוצים לנהל ולהסתכל על הנתונים. לרוב הארגונים והמשתמשים אין את הזמן ואת הסבלנות הנדרשים להקמת מחסן נתונים ארגוני, כי הלחצים לקבלת מידע גבוהים ואין אפשרות להשקיע את המשאבים. בנוסף, התנאים והדרישות משתנים כל הזמן, נפח הנתונים של המחסן הארגוני עצומים וכל זה - מרפה את ידי המתחילים לעסוק בנושא. לפעמים מגיעים למסקנה שאין זה מעשי לבנות מחסן נתונים ארגוני עבור ארגונים גדולים שבהם יש שונות גדולה של מוצרים ושירותים.

בגלל קשיים אלה נטשו, לפחות באופן זמני, חלק מהארגונים את הגישה הזו והגדירו מטרה צנועה יותר: מחסן נתונים מחלקתי, או מחסן נתונים נושאי ובפשטות - מרכול נתונים.

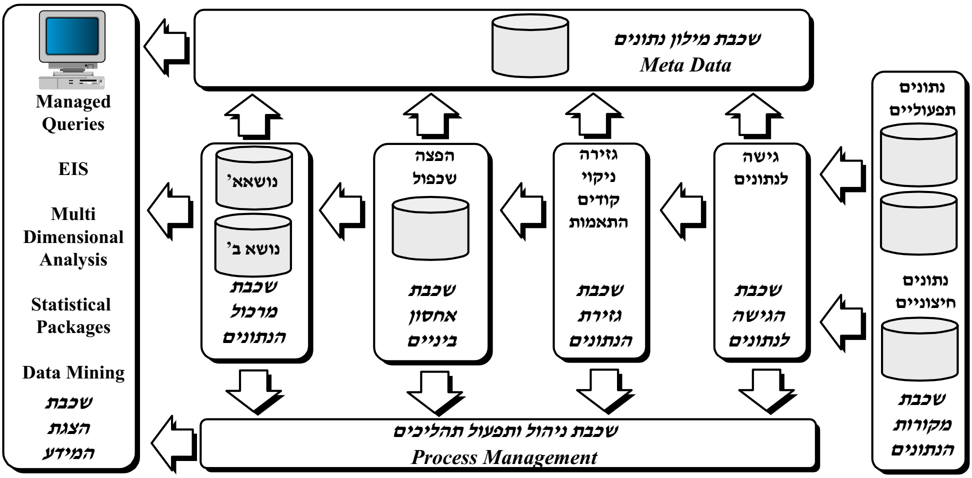
ארכיטקטורת מרכול הנתונים (Data Mart Architecture)

מרכול הנתונים הוא הרחבה של תפיסת מחסן הנתונים הארגוני לרמה המחלקתית, ולכן הוא ממוקד בנושא או בתחום מסוים ומוגדר. באופן טבעי, הוא קטן יותר ממחסן הנתונים הארגוני.

מרכול הנתונים (Data Mart) הוא מחסן נתונים שנועד לשרת, בדרך כלל, מחלקה אחת (או מספר מצומצם של מחלקות) וממוקד בנושא אחד בלבד.

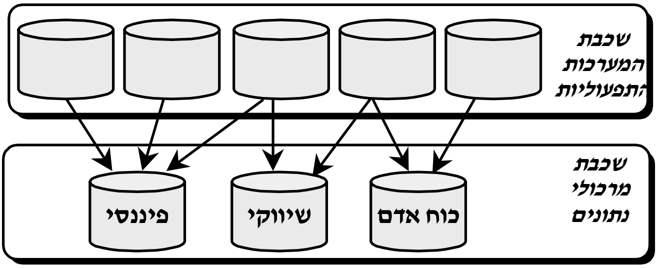
מבחינת תכולת הנתונים, מרכול הנתונים הוא נגזרת חלקית כלשהי של המחסן הארגוני. הוא מיועד לשימוש מחלקה אחת או מספר מצומצם של מחלקות קשורות ולרוב הוא ממוקד בנושא אחד. אחד מפרקי הספר מוקדש לנושא זה.

ארכיטקטורת מרכול הנתונים דומה לארכיטקטורת מחסן הנתונים הארגוני, בהבדל אחד מהותי: בעוד שיש רק מחסן נתונים ארגוני אחד, מרכולי נתונים יכולים להיות רבים.



תרשים 2.3: ארכיטקטורת מרכול נתונים.

ניתן לראות בתרשים 2.3 שכל השכבות שקיימות במחסן הארגוני קיימות גם כאן. מכיון שגם בארכיטקטורה זו קיימת אבחנה בין שכבת הנתונים התפעוליים לבין שכבת הנתונים המיועדים לתמיכה בקבלת החלטות, ארכיטקטורת מרכול הנתונים היא גם ארכיטקטורה דו-שכבתית מבחינת הנתונים.



תרשים 2.4: שכבות בארכיטקטורת מרכולי הנתונים.

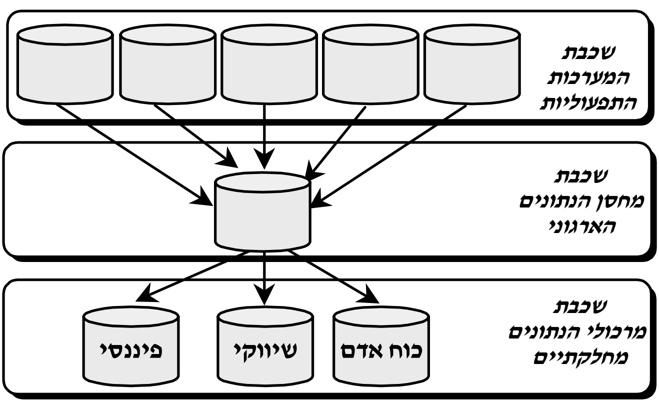
מחסן נתונים רב-שכבתי (Multi Tier Data Warehouse)

שתי הארכיטקטורות שהצגנו - המחסן הארגוני ומרכול הנתונים - אינן מיועדות להחליף זו את זו. הן ארכיטקטורות משלימות ומשולבות.

מחסן נתונים רב-שכבתי משלב בתוך ארכיטקטורה אחת מחסן נתונים ארגוני יחד עם מספר בלתי מוגבל של מרכולי נתונים.

להבדיל מהארכיטקטורה הדו-שכבתית של מרכולי הנתונים השואבים את הנתונים באופן ישיר מהמערכות התפעוליות, בארכיטקטורה הרב-שכבתית שואבים מרכולי הנתונים את הנתונים ממחסן הנתונים הארגוני. כל מרכול נתונים מכיל חלק מסוים מנתוני המחסן הארגוני. תהליכי הגזירה ליצירת המרכול מתבססים, בדרך כלל, על טכנולוגיות **שכפול נתונים** (Data Replication), הנתמכות כיום על ידי רוב בסיסי הנתונים היחסיים המודרניים.

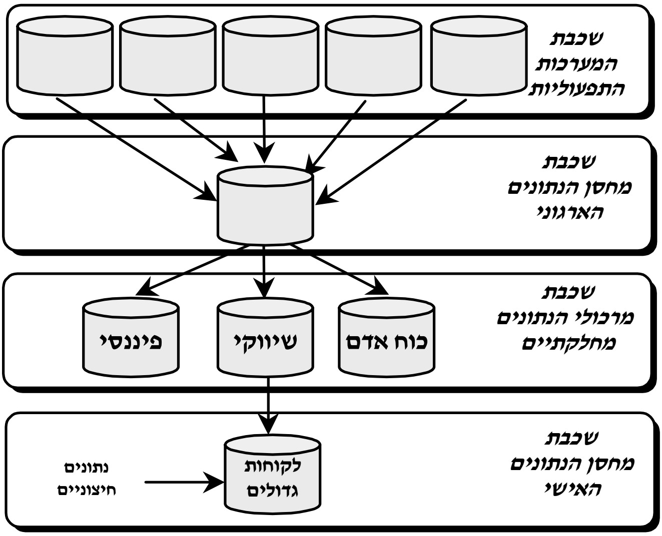
ארכיטקטורת מחסן הנתונים הרב-שכבתי היא המומלצת ביותר. מצד אחד היא מבוססת על מחסן נתונים ארגוני ומצד שני היא נותנת פתרונות למחלקות ולנושאים ומספקת כלים מיוחדים כגון **כלי ניתוח רב מימדיים** (OLAP) או **כלי כריית נתונים** (Data Mining). חלק מכלים אלה אינם פועלים באופן ישיר על בסיס הנתונים הטבלאי. לדוגמה, עבור כלי OLAP דרוש מבנה קוביה רב-מימדית, בשעה שכלי כרייה אחרים יכולים להסתפק במבנה של קובץ שטוח.



תרשים 2.5: ארכיטקטורת מחסן נתונים רב-שכבתי.

ארכיטקטורת מחסן הנתונים הרב-שכבתי משלבת את היתרונות שבקיום מחסן נתונים כלל ארגוני ואת היתרונות שברמת עצמאות והתאמה לדרישות ייחודיות של מחלקות שונות. הבעיה היא בקושי שבבניית המחסן הארגוני ובתפעול שוטף של ארכיטקטורה מבוזרת ומורכבת יחסית.

לפעמים ניתן לבנות גם מחסן נתונים המבוסס על שכבה נוספת: שכבת הנתונים המנוהלים במחשב האישי של משתמש מסוים. בארגון קיימים מספר מועט של משתמשים המבצעים ניתוחי מידע מורכבים ומאוד ייחודיים ומסיבות שונות, כגון סודיות או שימוש בכלי ניתוח מאוד ייחודיים, עדיף לגזור את הנתונים ממרכול מחלקתי (או לפעמים ישירות מהמחסן הארגוני) ולהעבירם לתחנת העבודה האישית.



תרשים 2.6: מחסן נתונים ארבע-שכבתי.

תרשים 2.6 מציג דוגמה של שכבה נוספת המופעלת במחשב אישי של משתמש מסוים. שכבה זו מקבלת נתונים ממרכול הנתונים השיווקי על ידי גזירת הלקוחות הגדולים בלבד וביצוע ניתוחים מיוחדים המשלבים גם נתונים חיצוניים.

ארכיטקטורת מאגר נתונים תפעולי (ODS - Operational Data Store)

בנוסף לדרישות ניתוח מידע עבור דרג מקבלי ההחלטות, הארגון צריך לספק מידע באופן שוטף גם לדרג התפעולי. אחת האפשרויות היא לספק מידע זה ישירות מתוך המערכות התפעוליות. אפשרות אחרת היא להקים ליד המערכת התפעולית מאגר נתונים ייעודי לאספקת מידע לדרגים התפעוליים. ברמת הארכיטקטורה קיים דמיון רב בין מחסן הנתונים לבין מאגר הנתונים התפעולי, אולם כפי שנראה מייד, האופי, עדכניות הנתונים המנוהלים בו וכלי הדיווח - שונים.

מאגר נתונים תפעולי הוא מחסן נתונים המיועד לשרת את הדרג התפעולי בארגון בכל הקשור לדיווחים ולניתוחי מידע שוטפים.

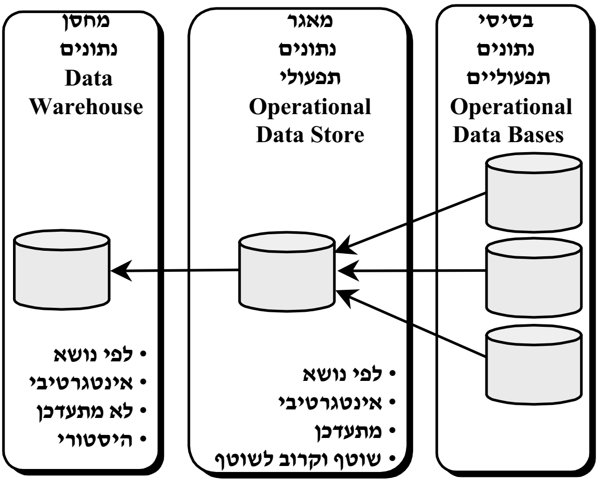
מאגר הנתונים התפעולי מיועד להפקת דיווחים וניתוחי מידע עבור הדרג התפעולי, ואולי גם טקטי, אבל עם דגש חזק על הפעילות השוטפת בארגון. נסקור את נקודות הדמיון והשוני בין מחסן הנתונים לבין מאגר הנתונים התפעולי.

- **ארגון לפי נושאים** - בדומה למחסן הנתונים, גם במאגר הנתונים התפעולי הנתונים מאורגנים לפי נושאים, ולא לפי תהליכים עסקיים, כמקובל במערכות תפעוליות.

❖ **מאגר אינטגרטיבי** - המאגר אוגר נתונים ממערכות תפעוליות שונות ופותר סתירות שונות.

❖ **מתעדכן** - להבדיל ממחסן נתונים המתעדכן רק דרך תהליכי טעינה, מקובל לעדכן את מאגר הנתונים התפעולי גם באמצעות תנועות המערכת התפעולית, במידת הצורך. מכיון שמאגר הנתונים התפעולי צריך להיות ברמת עדכניות גבוהה, לא תמיד תהליכי טעינה באצווה מספקים את דרישות הדרג התפעולי. יחד עם זאת, רצוי לבנות מאגר נתונים תפעולי המתעדכן באופן דומה למחסן הנתונים, במידה ורמת העדכניות מאפשרת.

❖ **שוטף וקרוב לשוטף** - להבדיל ממחסן הנתונים המנהל נתונים מפורטים לפרקי זמן ארוכים, העומק ההיסטורי של מאגר קצר יותר, כי הוא משרת בעיקר את הדרג התפעולי. בדרך כלל, מנוהלים בו הנתונים השוטפים והקרובים לשוטף (Near Current).



תרשים 2.7: ארכיטקטורת מאגר נתונים תפעולי ליד מחסן נתונים.

כפי שניתן לראות מתרשים 2.7, מאגר הנתונים התפעולי יכול לשמש כמאגר ביניים לפני העברת הנתונים אל מחסן הנתונים, אם כי אין זה תנאי הכרחי. יתרון השימוש במאגר הנתונים התפעולי כמאגר ביניים הוא בכך שתהליכי הגזירה וטיוב הנתונים מתבצעים בעת הכנסת הנתונים אליו, לכן אין צורך לחזור על הפעולה בעת ההעברה למחסן הנתונים.

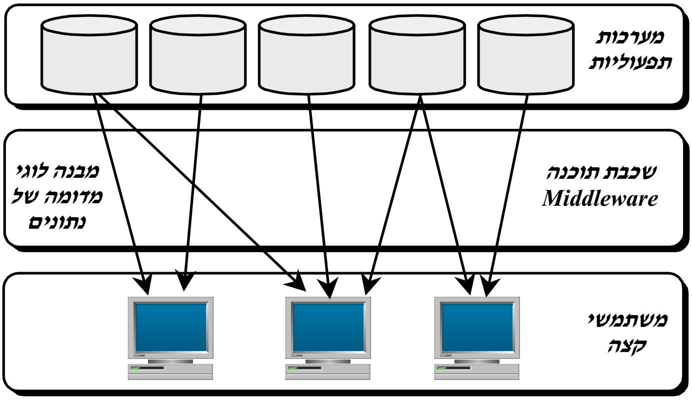
להבדיל ממחסן הנתונים, שצריך להכיל כלי ניתוח מידע מתוחכמים מאוד המיועדים לשימושם של משתמשים מתמחים, מאגר הנתונים התפעולי מכיל בדרך כלל מחוללי דוחות ושאילתות, מכיון שדרישות ניתוח המידע צנועות בהרבה.

מחסן נתונים מדומה (Virtual Data Warehouse)

למרות ההבדלים בין מערכות תפעוליות לבין מערכות לאספקת מידע, עדיין יש ארגונים המנסים ליישם תפיסה של מחסן נתונים מדומה.

מחסן נתונים מדומה הוא אוסף של כלי גישה ותוכנות קישור, המאפשרים גישה נוחה לנתונים תפעוליים מתוך כלי הצגת וניתוח מידע פשוטים יחסית.

הרעיון הבסיסי עליו מבוסס **מחסן הנתונים המדומה** הוא לאפשר גישה נוחה וקלה של המשתמשים לבסיסי הנתונים התפעוליים בארגון.



תרשים 2.8: ארכיטקטורה של מחסן מדומה.

בארכיטקטורה זו ניתן דגש רב על בניית **שכבת ביניים** (Middleware) המאפשרת קישור בין אוסף כלי קצה לבסיסי הנתונים התפעוליים בארגון. שכבת ביניים זו, הפועלת לפעמים על שרת ייעודי, מבוססת על תקנים כגון ODBC, DRDA, או על תוכנות ביניים מיוחדות המאפשרות גישה לבסיסי נתונים כמו IMS, IDMS, או לקבצי VSAM ואחרים.

כפי שניתן להניח, ארכיטקטורה זו מפתה מאוד, מפני שאין צורך בגזירת נתונים וניהול בסיסי נתונים נפרדים עבור מערכות אספקת המידע. מתקינים אוסף של כלי קצה ותוכנות קישור מסוימות, והרי לנו מחסן נתונים. איך בכלל ניתן לעמוד בפני פיתוי שכזה? הבעיה היא שתפיסה זו פשוט "לא עובדת", ויש לכך סיבות רבות: הבדלים במבנה הנתונים, סתירות בין הנתונים, עומס על מחשבי הייצור, חוסר יכולת לבצע אינטגרציה בין נתונים המנוהלים במערכות מחשוב שונות ורמת הסיבוכיות שהמשתמש נחשף לה. כל אלו הן רק חלק מהבעיות שארגון המנסה לאמץ תפיסה זו עלול להיתקל בהן.

ניתן לומר שארכיטקטורה זו מתאימה כפתרון ביניים בלבד לארגונים הנתקלים בקשיים בבניית מחסן נתונים אמיתי והמבקשים להקל, לפחות באופן זמני, על חלק מהמשתמשים על ידי מתן אפשרות קלה יותר של גישה לנתונים. "ארכיטקטורה" פשטנית זו קיימת רק בעלוני השיווק של ספקים מסוימים, המבקשים למכור את האשליה שניתן לבנות מחסן נתונים במהירות ובקלות.

סיכום

בפרק זה סקרנו את הארכיטקטורות השונות של מחסני נתונים: מחסן הנתונים הארגוני, מרכול הנתונים, מחסן הנתונים הרב-שכבתי, מאגר הנתונים התפעולי ומחסן הנתונים המדומה.

אחת השאלות העיקריות שהארגון צריך לשאול את עצמו מייד עם תחילת יישום מחסן נתונים היא **איזו ארכיטקטורה** ברצונו לממש. התשובה לשאלה זו מורכבת, ותלויה בארגון עצמו. אנשי המקצוע העוסקים במחסני הנתונים חלוקים בדעותיהם באשר לארכיטקטורה המומלצת. יש הטוענים בלהט רב שרק המחסן הארגוני יביא את התועלות לארגון, בעוד שאחרים טוענים באותו להט שמימוש מחסן נתונים ארגוני נדון לכישלון ואינו ישים, בשל חוסר היכולת לספק את כלל הצרכים והדרישות בבסיס נתונים אחד ויחיד. לכן, הם טוענים, רק מרכול הנתונים נותן תשובה לצרכי הארגון. התשובה הנכונה היא, כמובן, שהארכיטקטורה המומלצת היא המחסן הרב-שכבתי, המכיר הן בצורך לבניית מחסן נתונים מרכזי והן בצורך לתת מענה לדרישות המשתמשים, המחלקות וכלי הניתוח השונים על ידי בניית מרכולי נתונים. בעיה לא פשוטה היא כיצד מקימים מחסן נתונים רב-שכבתי. נדחה את הדיון בנושא זה לאחד הפרקים הבאים.

חשוב להדגיש שדעתנו המלומדת לא תמיד קובעת, אלא התקציב העומד לרשות הארגון מכתיב לא פעם את ארכיטקטורת מחסן הנתונים שלו. יישום ארכיטקטורת מרכול הנתונים זולה יותר וקצרה יותר, ולכן זו החלופה המועדפת על ידי ארגונים המבקשים לצמצם את השקעותיהם בטכנולוגיית המידע. התקציב הנמוך יחסית, לוח הזמנים הקצר וקלות היישום, הם בלי ספק הגורמים העיקריים לכך שארגונים רבים בוחרים בחלופה זו.

הקמת מספר מרכולי נתונים מחלקתיים לא-עקביים גורמת לארגון לאבד יתרון אחד חשוב - היכולת לקבל נקודת מבט כוללת על כל הנתונים. כל מרכול נתונים מנהל רק חלק מהנתונים, ומכיון שגם לא טורחים תמיד לתאם בין המחסנים השונים, הארגון נחשף מחדש לבעיות חוסר עקביות בין הנתונים, סתירות בין נתונים (הנובעים ממרכולי נתונים שונים) וכדומה. נדגיש גם שמאמצי הגזירה מהמערכות התפעוליות, ומאמצי הניקוי והטיוב של הנתונים מתבצעים מספר פעמים, תוך בזבוז משאבי פיתוח ומשאבי מחשוב.

ממחקרי שוק שונים מסתבר שיישום מרכול נתונים יכול להתבצע תוך מספר חודשים (פחות מחצי שנה) ובעלות של מספר מאות אלפי ש״ח ועד כמיליון ש״ח. לעומת זאת, משך הקמת מחסן נתונים ארגוני יכול לקחת עד כשנתיים ועלותו מספר פי עשר ויותר. בשל הבדלים אלה ארגונים רבים מעדיפים להתחיל ב**יישום ניסיוני** (Pilot) של מרכולי נתונים, כשלב מקדים לבניית מחסן הנתונים הארגוני. זו תפיסה נבונה המאפשרת התנסות הדרגתית הן בצד התועלות והן בצד הקשיים הצפויים. ארגונים אלה מנסים להגדיר ארכיטקטורה אחידה, סטנדרטים ועקרונות, כדי להבטיח שבבוא העת ניתן יהיה לשלב את מרכולי הנתונים עם מחסן ארגוני, לקבלת ארכיטקטורה משולבת רב-שכבתית.

פרק 3: סקירת מרחב הטכנולוגיות לניתוח מידע

מבוא

מחסני הנתונים, על פי הגדרתם, נועדו לתמוך בניתוח מידע לתמיכה בתהליכי קבלת החלטות ולאפשר לארגונים לנצל טוב ביותר אחד מהמשאבים החשובים שלו - **המידע**. כפי שכבר הוסבר, מחסן הנתונים אינו מוצר, אלא תפיסה כוללת המשלבת מספר מוצרים לבניית פלטפורמה ארגונית לניהול וניצול יעיל של מידע למטרות עסקיות. בסופו של דבר, מחסן הנתונים נועד להעמיד את הנתונים והמידע לרשות המשתמשים. ייעודו העיקרי הוא לאפשר להם לנתח ולחקור את הנתונים כדי להבין ולפענח טוב יותר את ביצועי הארגון ולדעת מהם הגורמים המשפיעים על ביצועים עסקיים אלה. כל זאת, כדי לאפשר להם לקבל החלטות טובות ומבוססות יותר.

ניתוח נתונים (Data Analysis) הוא נושא רחב ביותר ונפוץ מאוד בתחומי המחקר האקדמי, ההנדסי והשימושי. בפרק זה נתמקד בתהליכי ניתוח מידע בסביבת מחסן הנתונים. כלומר, בתהליכי ניתוח מידע עסקי שמטרתם פתרון בעיות עסקיות ותמיכה בקבלת החלטות עסקיות. תהליכים אלה יכולים להתחיל בתהליכי ניתוח מידע פשוטים מאוד ולהגיע לתהליכי ניתוח מידע מורכבים ביותר, בהתאם לאופי ולסוג ההחלטות שיש לקבל. בכל מקרה, הרכיב האנליטי של מחסן הנתונים חייב לתמוך בגישה מהירה לנתונים, בבניית מיגוון שאילתות מזדמנות וקבועות, בביצוע חישובים שונים על הנתונים, בחיזוי ובתכנון, בסימולציות What If וכדומה. כפי שנראה בפרק זה, הרכיב האנליטי של מחסן הנתונים איננו הומוגני, אלא מורכב מאוסף כלים ומוצרים המספקים צרכים אנליטיים שונים ומגוונים.

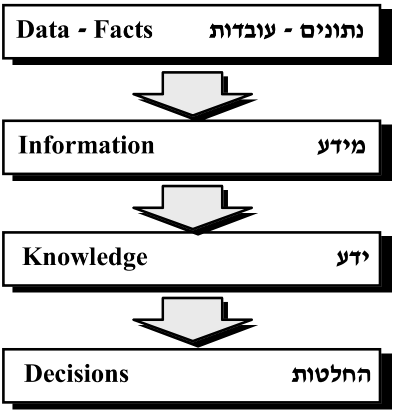
אחד המאפיינים החשובים של תהליכי ניתוח המידע הוא שהם בעלי אופי איטרטיבי ולא לינארי. הניתוח מתחיל מנקודה מסוימת, ועל סמך המידע המתקבל עולות שאלות חדשות, לעיתים בכיוונים חדשים לחלוטין. בחינת המידע המתקבל עשויה להוביל לשאלות וחקירות נוספות, וחוזר חלילה, עד למיצוי המידע הנדרש לקבלת ההחלטה. בגלל מיגוון תהליכי הניתוח האפשריים, אין שיטה אחת וטכנולוגיה אחת המסוגלים לתמוך בתהליך קבלת ההחלטות. לכן, סביבת מחסן הנתונים היא עתירת טכנולוגיות לניתוח מידע.

בפרק זה נרחיב את הדיון בדרכים לניתוח המידע ונגדיר את המושג **ניתוח מידע**. נסקור את התהליך בו נתונים הופכים בסופו של דבר להחלטות, נציג שתי גישות שונות לניתוח מידע - גישת האימות וגישת הגילוי, נסקור את משימות הניתוח השונות המתבצעות על ידי גורמים שונים בארגון, ונסכם במפה המציגה את מיגוון הטכנולוגיות השכיחות בסביבת מחסן הנתונים.

מנתונים דרך מידע ועד להחלטות

מהות תפיסת מחסן הנתונים היא תמיכה בהחלטות. לכן יש חשיבות בהבנת ״שרשרת המזון״ המתחילה בנתונים גולמיים ומסתיימת בהחלטות עסקיות. ניתן להסתכל על תהליך זה גם כעל **שרשרת ערך לתמיכה בקבלת החלטות (Decision Support Value Chain)**. שרשרת בה מתבצעות המרות שונות המגדילות באופן מתמיד את ערך הנתונים, עד להפיכתם להחלטות עסקיות. מכיון שהמושגים נתונים, מידע, ידע והחלטות ישמשו אותנו תדיר בהמשך ננסה להגדיר אותם במדויק, אם כי לא באופן פורמלי ומתמטי, אלא בהגדרה שימושית.

נתחיל את הדיון בהצגת התהליך ההופך נתונים להחלטות.



תרשים 3.1: שרשרת ההמרות מנתונים להחלטות.

כפי שניתן לראות מתרשים 3.1, התהליך מתחיל בנתונים גולמיים, ההופכים למידע ההופך לידע שבאמצעותו אנשים מקבלים החלטות. בכל שלב מתבצעת המרה כלשהי, שיכולה להיות פשוטה מאוד ולעיתים מורכבת מאוד. כאן המקום להדגיש שתהליך העלייה בערך או ההתקדמות בתוך שרשרת התהליך, אינו לינארי אלא איטרטיבי. כלומר, התהליך כולו מבוסס על רצף המתחיל בדרך כלל מהנחות מסוימות, ממשיך בהפעלת אוסף שאילתות על הנתונים, עובר לניתוח הממצאים וחקירת הנתונים, משם - להפעלת שאילתות נוספות וחוזר חלילה. כך, באופן איטרטיבי, עד לקבלת תמונת המידע והידע הנדרשים לקבלת ההחלטות.



תרשים 3.2: תהליך איטרטיבי בתוך השרשרת.

להבהרת המושגים, התהליך וההגדרות נשתמש בדוגמה העוסקת במכירת מכוניות, ונלווה את השרשרת מתחילתה ועד סופה.

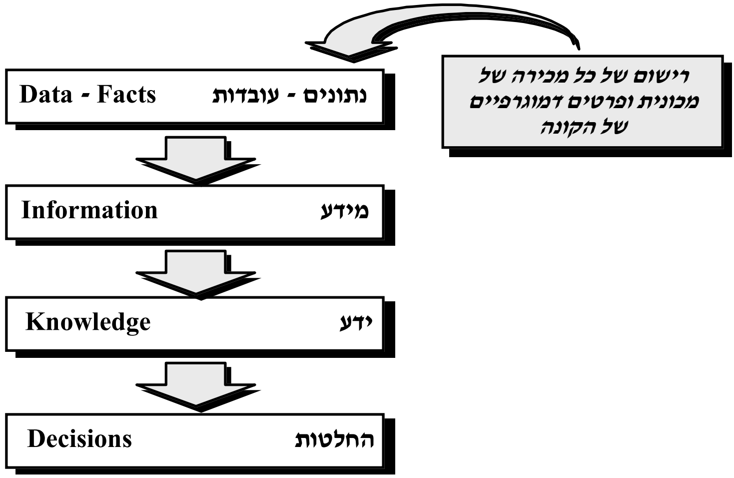
נתונים (Data)

נתונים הם אוסף של עובדות בסיסיות על אנשים, אירועים ויישויות הרלוונטיים לארגון וליישום. הנתונים נאגרים ומעובדים במערכות המידע התפעוליות של הארגון.

דוגמאות לנתונים: מחיר פריט, שם לקוח, משך שיחת טלפון, שם מו"ל של ספר, כתובת ספק הציוד, יתרה במלאי של פריט, כמות שהוזמנה, היתרה בחשבון הבנק, תאריך התנועה האחרונה בחשבון הבנק וכדומה. במובן מסוים ניתן להתייחס לנתונים כאל אלמנטים אטומיים במודל הנתונים הכולל של הארגון. בעקרון, זו הרמה הנמוכה שמעניינת את הארגון. לדוגמה, לספרה השנייה ביתרת המלאי של הפריט אין כל חשיבות כשהיא לעצמה, אלא רק כחלק מהמספר הכולל המייצג את היתרה כולה. בדומה, לא מעניין אותנו מהי האות השביעית בשמו של הלקוח, כי מעניין אותנו שמו המלא בלבד.

מערכות המידע התפעוליות קולטות את העובדות הבסיסיות הללו, כלומר את הנתונים, באמצעות **תנועות** (Transactions), מאחסנות אותן בבסיסי הנתונים ולאחר מכן מאפשרות את עיבודן. מערכות המידע המודרניות מסוגלות כיום לקלוט ולעבד כמויות עצומות של נתונים.

אם נתייחס למערכת המבצעת רישום של מכירת מכוניות בסוכנות למכירת כלי רכב, נראה שהנתונים הבסיסיים הנאגרים על ידה יהיו למשל: שם לקוח, כתובת, גיל, רמת הכנסה, סוג הרכב שקנה, מחיר הרכב, תאריך הרכישה, שם יצרן הרכב, דגם, מספר הרישוי, נפח מנוע, צבע המכונית, סוג תיבת ההילוכים וכדומה.



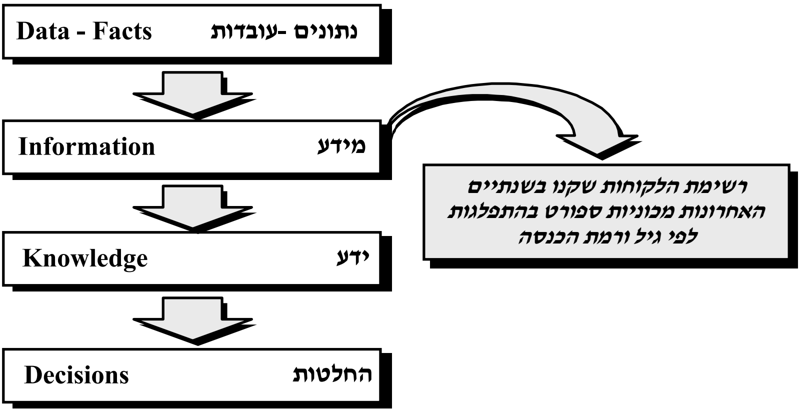
תרשים 3.3: שלב רישום הנתונים.

מידע (Information)

מידע מייצג את הנתונים שעברו עיבוד ושיוך בהקשר מסוים. המידע מוצג בצורה הנוחה לבני אדם: דוחות, גרפים, טבלאות וכדומה.

המרת נתונים למידע מתבצעת בדרכים שונות: מיון הנתונים בסדר מסוים, שליפת נתונים העונים על קריטריון כלשהו, עיבודים וחישובים מתמטיים כלשהם (סיכומים, ממוצעים, הפעלת נוסחאות וכדומה), הצגת הנתונים בצורה של גרף, הדגשה של ערכים חריגים וכדומה.

אם נמשיך בדוגמה שהבאנו מסוכנות כלי הרכב, נוכל לעבד את הנתונים הגולמיים ולהפיק מהם סוגי מידע שונים. נניח שמנהל הפרסום צריך לקבל החלטות לבניית התדמית למכוניות שונות שהסוכנות מוכרת. זו למעשה הגדרת הבעיה וממנה מתחיל תהליך חקירת הנתונים. כחלק מתהליך זה נוכל להציג את התפלגות מכירת סוגי כלי הרכב לפי חתך גיל הקונה ורמת הכנסתו, את מספר כלי הרכב שנמכרו במהלך כל אחד מ- 24 החודשים האחרונים, כמה כלי רכב של יצרן מסוים נמכרו לקונים הגרים באזור מסוים בארץ וכדומה. כפי שניתן לראות, על בסיס הנתונים הגולמיים שנאגרו במערכת התפעוליות של סוכנות הרכב ניתן להפיק אינספור חתכי מידע שיכולים לשמש משתמשים שונים לביצוע תפקידם ולניתוח ביצועי העסק. התהליך הוא איטרטיבי ולכן לאחר הצגת הנתונים בצורה או ברמה מסוימת, סביר להניח שנרצה להוסיף ולחקור בכיוון מסוים כתוצאה מהמידע שהתקבל.



תרשים 3.4: שלב המרת נתונים למידע.

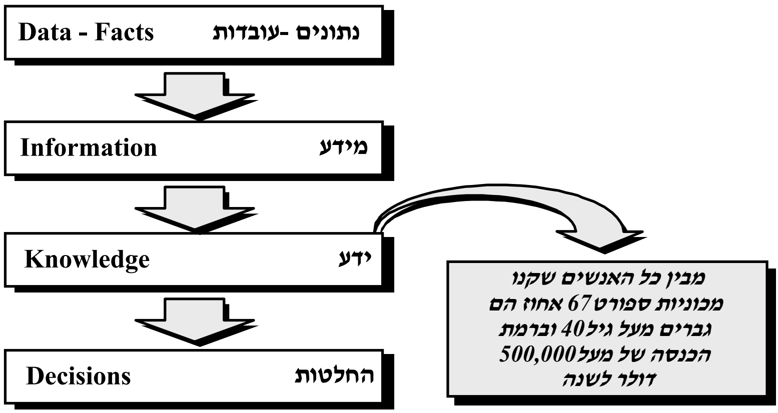
מחסן הנתונים הוא אחד המרכיבים העיקריים בתהליך ההמרה של הנתונים למידע, מכיון שהוא מאחסן את מירב הנתונים התפעוליים במבנה ייעודי ומאפשר את המרתם למידע באמצעות מיגוון כלים. כלי העבודה של מחסן הנתונים הם מחוללי שאילתות ודוחות, כלים גרפיים להצגת המידע, כלים סטטיסטיים לניתוח הנתונים וכדומה. הכלים מאפשרים לבצע את החקירה והצגת המידע בצורה נוחה וידידותית.

ידע (Knowledge)

ידע הוא תוספת שניתנת לנו להבנה של נושא מסוים.

תהליך ההמרה של המידע לידע מתרחש במוחו של האדם ומבוסס על ההתבוננות במידע וניתוחו. קשה להגדיר תהליך זה כ**תהליך מובנה**, מכיון שהוא מבוסס לא מעט גם על גורמים כגון ניסיון, הבנת המידע המוצג וגורמים נוספים.

בדוגמה שלנו, מתוך התבוננות ברשימת רוכשי מכוניות הספורט בשנתיים האחרונות לפי גיל ורמת הכנסה, נוכל להגיע למסקנה שרובם גברים מעל גיל 40 ובעלי רמת הכנסה של מעל 500,000 ש"ח בשנה. זהו ידע חדש שנוצר במהלך ניתוח הנתונים, הפיכתם למידע וניסיון להבין ממנו משמעויות נסתרות, או גלויות.



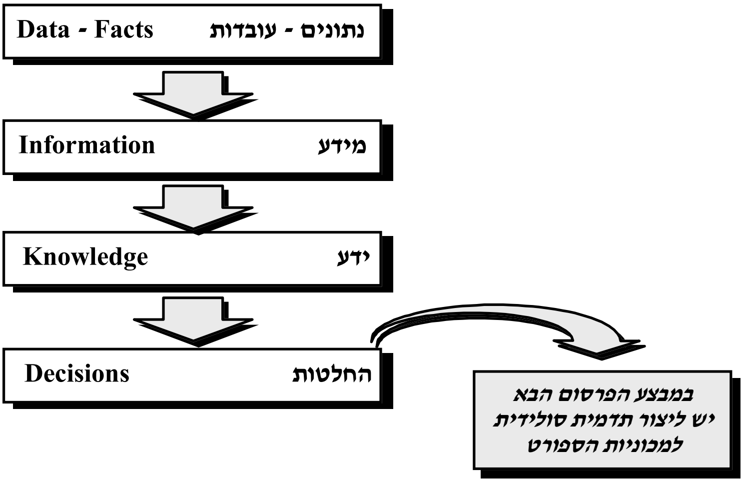
תרשים 3.5: שלב המרת המידע לידע.

תהליך הפיכת המידע לידע יכול להיות אנושי ולהתבצע במהלך חקירת הנתונים והמידע, או שהוא יכול להיות תהליך ממוחשב המבוסס על שימוש באלגוריתמים שונים המופעלים על הנתונים כדי להסיק מהם את הידע. טכניקות ממוחשבות אלו ידועות כיום בשם הכולל **כריית נתונים** (Data Mining).

החלטות (Decisions)

כל ארגון מבקש לקבל החלטות מושכלות ומבוססות במטרה להפוך החלטות למעשים ופעולות שתשפענה בדרך כלשהי על הארגון. הבעיה היא שהחלטות מבוססות על מספר רב מאוד של פרמטרים וחשוב לא פחות, הן אינן מתבססות רק על ידע, אלא גם על ניסיונם של המנהלים, הערכותיהם ועל הבנת המשמעות של המידע המובא בפניהם. אותו מידע המוצג לאנשים שונים יכול להפוך לידע שונה, ובסופו של דבר להביא להחלטות שונות. כל זה נובע מהבנה שונה ומהקניית משמעות שונה למידע על ידי כל מנהל. כתוצאה מתהליך רכישת הידע ניתן לעבור לשלב הבא: הפיכת הידע להחלטות אופרטיביות.

בהמשך לדוגמה שלנו, כתוצאה מהידע שרוב רוכשי מכוניות הספורט הם גברים מעל גיל מסוים ובעלי רמת הכנסה מסוימת, ניתן להגיע להחלטות שונות. למשל, מנהל הפרסום יכול להגיע להחלטה לנסות ולבנות תדמית סולידית יותר למכוניות הספורט, לפתוח במבצע מכירות המותאם לפלח אכלוסיה זה וכדומה. כמובן, שאם היינו מציגים ידע זה למנהל השיווק בחברה המייצרת את המכוניות, הוא יכול היה להגיע להחלטות שונות בדבר צורת המכונית, האבזור הפנימי, טווח המחירים הסביר למכוניות אלו וכדומה.



תרשים 3.6: שלב המרת הידע להחלטות.

מה זה ניתוח נתונים

את הדיון בנושא ניתוח המידע נתחיל בהבחנה בין התהליכים השונים המתבצעים בארגון. מקובל לחלק את התהליכים לשתי קטגוריות עיקריות: תהליכים תפעוליים ותהליכים אנליטיים.

- **תהליכים תפעוליים (Operational Processes):** אוסף התהליכים הרגילים שבאמצעותם הופכים אוסף של תשומות לאוסף של תפוקות. למשל, אם הארגון הוא יצרני, הפעילויות השוטפות הן פיתוח ותכנון המוצרים, רכישת חומרי גלם, אחסון חומרי הגלם במחסנים, ייצור המוצרים, קבלת הזמנות מהלקוחות, משלוח המוצרים ללקוחות, תיקון תקלות במוצרים וכדומה. אם הארגון עוסק במתן שירות כגון בנק, או חברת ביטוח, הפעילויות השוטפות הן לנהל סניפים בהם הלקוחות מקבלים את השירות, להגדיר שירותים חדשים, למכור שירותים ללקוחותיהם, לחייב את הלקוחות בגין בעמלות השירותים, לספק ללקוחות דיווח מצב החשבון או הפוליסה וכדומה. אם החברה עוסקת בטלקומוניקציה הפעילויות השוטפות הן תכנון ופיתוח רשת תקשורת, התקנת מרכזיות ציבוריות או ציוד תקשורת נתונים מתאים, פיתוח ושיווק שירותים, רכישה והתקנה של ציוד חדש, תחזוקת רשת התקשורת, חיוב לקוחות בגין צריכת שירותים, טיפול בתקלות רשת וכדומה.

- **תהליכים אנליטיים (Analytical Processes):** תהליכים אלה מבוצעים על ידי הארגון ומטרתם ניתוח נתונים לצורך הבנת התנהגות הארגון. תהליכים אלה עוקבים אחר התפתחויות בפעילות הארגון במטרה לקבל החלטות עסקיות. מטרת ההחלטות העסקיות היא לשנות את ביצועי הארגון ברמה האופרטיבית השוטפת, ברמה הטקטית לטווח הבינוני, וגם ברמה האסטרטגית בטווח הארוך יותר. את התהליכים האנליטיים מבצעים מספר רב של גורמים בארגון: מנהלים בדרג זוטר, מנהלים בדרג בכיר, מנתחי מידע מקצועיים ואחרים.

מערכות עיבוד תנועות (OLTP) הן מערכות מידע תפעוליות שהתפתחו במשך השנים במטרה לתמוך בביצוע התהליכים התפעוליים השוטפים בארגון. באופן טבעי, הן משרתות בעיקר את הדרג התפעולי של הארגון, שעוסק בביצוע הפעילויות השוטפות והיומיומיות: עובדי הייצור, קניינים, מחסנאים, מנהלי חשבונות, אנשי שירות וכדומה. הפעילויות השוטפות מתאימות מאוד לטכנולוגיית המחשב, מכיון שהן פשוטות וחוזרות על עצמן. שימוש במערכת מידע ממוחשבת יכול להביא לייעול ופישוט תהליכים אלה.

ניסיונם של המנהלים ומנתחי המידע להשתמש במערכות תפעוליות כמערכות התומכות גם בתהליכים האנליטיים, נתקל בקשיים בשל האופי השונה לחלוטין של תהליכים אלה. עם השנים החל להתפתח סוג חדש של מערכות ממוחשבות - מערכות לניתוח נתונים מקוון. מערכות אלו יועדו לאוכלוסיה שונה: עובדי ידע, מנהלים ומקבלי החלטות בארגון. הופעת המחשבים האישיים ושרתים זולים ועתירי עוצמת עיבוד, והוזלת אמצעי האחסון ורשתות התקשורת המהירות היו גורם עיקרי בהתפתחויות שחלו בטכנולוגיית המידע. כל אלה שימשו כפלטפורמה שאפשרה את התפתחות המערכות הממוקדות בכל סוגי הפעילויות האנליטיות לניתוח הנתונים המתבצע בצורה אינטראקטיבית. המערכות התומכות בניתוח הנתונים זכו לשם כולל ונפרד - מערכות תומכות החלטה. הדבר מדגיש את האבחנה בינן לבין המערכות התומכות בפעילות השוטפת של הארגון. תחילה נציג הגדרה פורמלית עבור הפעילות האנליטית של ניתוח נתונים בסביבת מחסן הנתונים.

ניתוח נתונים בסביבת מחסן נתונים מתייחס לכל הפעילויות האנליטיות לניתוח וחקר הנתונים כדי לאפשר החלטות עסקיות ופתרון בעיות. פעולות אלו מתבצעות בצורה אינטראקטיבית עם המחשב ומשתמשים בהן במיגוון טכנולוגיות, כלים ושיטות וביניהם: מחוללי שאילתות, כלים גרפיים להצגת נתונים, כלי ניתוח רב-מימדי, טכנולוגיית כריית נתונים, כלים סטטיסטיים וכדומה.

נוכל לומר שניתוח הנתונים עוסק בפעילות המרה ההופכת את הנתונים למידע ולידע. כך, בסופו של דבר, יכול הארגון לקבל החלטות טובות יותר. התהליכים האנליטיים מתבצעים בעיקר על ידי אוכלוסיית המנהלים, מקבלי ההחלטות ומנתחי המידע. לכן הם ממוקדים פחות ברמה השוטפת היומיומית ויותר אלא ברמות גבוהות, תוך חיפוש אחר מגמות, תופעות חריגות ותובנות חדשות. התהליכים האנליטיים הם בעלי אופי שונה מתהליכים שוטפים ומבוצעים על ידי משתמשים שונים, ולכן טבעי שהם מבוססים על טכנולוגיות מידע שונות מאלו התומכות בתהליכים התפעוליים. תפיסת מחסן הנתונים היא התפיסה הכוללת העוסקת בהגדרה, בבנייה וביישום של מערכות ייעודיות, אשר מטרתן לתמוך בביצוע התהליכים האנליטיים של הארגון.

בקרב מנהלים ואנשי מקצוע קיים בלבול מושגים מסוים בכל הקשור למושגים **ניתוח נתונים**, **מערכות תומכות החלטות** ו**מחסני נתונים**. הנטייה היא לחשוב שכל אלה הם מושגים נרדפים, ולא כך הדבר. **מחסן הנתונים** מייצג תפיסה רחבה ושלמה, שמטרתה ניהול **וניתוח נתונים** ל**תמיכה בתהליכי קבלת החלטות**. הוא עוסק בכל קשת הפעילויות של איסוף הנתונים, גזירתם ממערכות תפעוליות או מקורות חיצוניים, שיפורם וטיובם, ארגונם בצורה מתאימה, טעינתם לבסיס נתונים ייעודי, אספקת הנתונים לכלי הניתוח, תפעול ותמיכה במשתמשי כלי הניתוח וכדומה.

במילים אחרות, **מחסן הנתונים** הוא תהליך קצה לקצה, בעוד ש**ניתוח הנתונים** מהווה רק חלק אחד מהמכלול הנקרא **מחסן נתונים**. אמנם, זהו החלק העיקרי והמטרה שלשמה נבנה מחסן הנתונים, אולם קיימת אבחנה ברורה בין המושג **מחסן נתונים** לבין המושג **ניתוח הנתונים**. חלק מהבלבול נובע מכך שניתוח הנתונים הוא חלקו הגלוי של הקרחון - זה מה שהמשתמשים במחסן הנתונים עושים. חלק גדול מפעילויות מחסן הנתונים הן הפעילויות התומכות, המסתתרות מתחת לפני המים וכוללות גזירת נתונים, בדיקות תקינות, טיוב נתונים, טעינת נתונים, בניית סיכומים, שיפור ביצועים, גיבוי נתונים וכדומה. דרך טובה להבין את האבחנה הזו היא להתייחס לפעילות ניתוח הנתונים כאל ה**פעילות הקדמית** (Front Office), הגלויה, בעוד שחלק גדול מתפיסת מחסן הנתונים עוסק בכל ה**פעילות האחורית** (Back Office), זו שמאחורי הקלעים, אבל בזכותה ניתן לבצע את פעילות ניתוח הנתונים.

סיווג תהליכי ניתוח הנתונים - אימות מול גילוי

מרחב התהליכים האנליטיים לניתוח הנתונים והמידע גדול ומתחיל בתהליכי ניתוח נתונים פשוטים מאוד ומגיע עד לתהליכי ניתוח מידע מורכבים ביותר. נשתמש במספר דוגמאות כדי להציג את רמת התחכום ההולכת וגדלה של פעילויות ניתוח המידע השונות. נתחיל בהצגת דוגמה לעיבוד אנליטי פשוט, נעבור לדוגמה מורכבת יותר ונסיים בדוגמה לעיבוד אנליטי מורכב.

- **עיבוד אנליטי ברמת מורכבות פשוטה:** נניח שמנהל בחברת טלקומוניקציה כלשהי מבקש לקבל מידע על פעילות לקוחות החברה בתקופה מסוימת ובאזור מסוים. לשם כך הוא יציג בוודאי שאילתה כגון "הצג את רשימת כל המנויים שהתקשרו ממספר טלפון באזור חיוג 03 למספרי טלפון באזור חיוג 04 בתאריך 10/2/99 בין שעות 10:00 ו- 10:30, ואשר משך השיחה היה מעל 10 דקות". במבט ראשון, במונחים של עיבוד אנליטי מקוון, נראית השאילתה מורכבת למדי, אך זו שאילתה פשוטה מאוד ובמובן מסוים היא רק תחילת הדרך בניתוח המידע. כאשר המנהל מקבל את התשובה לשאילתה זו הוא יוכל לבקש פירוט נוסף המתמקד בטווח זמן מצומצם יותר, למשל בין השעות 10:00-10:15, לבקש פירוט לתאריך שונה, או לבקש פירוט של שיחות קצרות יותר. עיבוד אנליטי כזה נתמך על ידי הטכנולוגיה של מחוללי שאילתות ודוחות פשוטים הפועלים על בסיס נתונים טבלאי. הם מאפשרים למשתמש לנסח את בקשתו, לקבל את התוצאה במהירות, להציג אותה באופן גרפי, לבקש פירוט נוסף, לשנות את הפרמטרים של השאילתה ובדרך איטראטיבית להגיע למידע המבוקש.

- **עיבוד אנליטי בעל רמת מורכבות בינונית:** נניח שהמנהל שפגשנו בדוגמה הקודמת מבקש לראות מיהן ארצות היעד הפופולריות ביותר לשיחות הטלפון. הוא יכול לנסח בקשה כזו לקבלת מידע: "מהן 5 הארצות אליהן היה מספר ההתקשרויות הרב ביותר החודש, ומהו נפח התנועה אל ארצות אלו החודש לעומת החודש הקודם".

דוגמה נוספת לתהליך אנליטי מורכב יותר: "הצג את סך ההכנסות לפי אזור חיוג, הצג פירוט לפי סוגי השירותים השונים ופירוט לפי חודשים לשלוש השנים האחרונות". מחוללי השאילתות הרגילים אינם תומכים בבקשות מידע אלו בצורה פשוטה. במשך השנים התפתחה טכנולוגיה ייעודית לניתוחי מידע מורכב מסוג זה המסוגלת להתמודד עם שאילתות כאלו, וגם מורכבות יותר. תפיסה זו נקראת **ניתוח רב-מימדי.**

- **עיבוד אנליטי ברמת מורכבות גבוהה:** עיבוד אנליטי מורכב יכול להיות, לדוגמה, תהליך תשובה לשאילתה מסוג "מה התחזית לגבי נטישת לקוחות, במידה ותעלה רמת התעריפים לשיחות בינלאומיות ליעדים מסוימים ב- 10%, ומה תהיינה השלכות השינוי על רווחיות החברה". כדי לענות על בקשה מסוג זה יש להבין מהם הפרמטרים המאפיינים לקוחות נוטשים, כגון נתונים דמוגרפיים, רמת הכנסה, כמות דקות שיחה, רגישות הביקוש לרמת התעריפים וכדומה. לאחר מכן יש לפתח מודל מתמטי כלשהו לחיזוי הנטישה. הטכנולוגיה של מחוללי השאילתות, ואפילו של כלי הניתוח הרב-מימדיים, מתקשים לתמוך בביצוע עיבודים אנליטיים כאלה. לכן התפתחו עם השנים טכנולוגיות מתקדמות התומכות בעיבודים אנליטיים מיוחדים אלה: **כריית נתונים**.

כפי שניתן לראות מדוגמאות אלו, התהליכים האנליטיים מגוונים מאוד ולכן טבעי הדבר שהם נתמכים על ידי טכנולוגיות שונות. דרך נוספת להבין את השוני בין תהליכי העיבוד האנליטי לבין הטכנולוגיות השונות היא לבחון את הנחת העבודה הבסיסית העומדת מאחורי כל אחת מהשיטות. מקובל לומר שקיימות שתי קטגוריות שונות של תהליכים אנליטיים: **תהליכים מבוססי אימות** ו**תהליכים מבוססי גילוי**.

- **תהליכים אנליטיים מבוססי אימות (Verification Based Analysis):** מספר רב של כלים אנליטיים מבוססים על העיקרון שהמשתמש בונה הנחה כלשהי, היפותזה, ומתחיל להפעיל טכניקות ניתוח שונות שאמורות לאושש, או לסתור אותה. תהליכים אלה מבוססים על המשתמש, על יכולתו להניח את ההנחות הנכונות, על שיקוליו בעת בחינת ההנחות ועל הפרשנות שלו לממצאים. במילים אחרות, תפיסה זו מבוססת לא מעט על כישוריו של המשתמש, האינטואיציה שלו, יכולתו להניח הנחות, לבנות את השאלות הנכונות ויכולתו לפרש נכון את הממצאים. יעילות השיטה תלויה במספר רב של גורמים, וביניהם יכולתו של מנתח המידע לחרוג "מהקופסה", כלומר לבחון גם תבניות ומגמות לא ידועות וגלויות. בהתבסס על אבחון זה, ניתן לומר שטכנולוגיית מחוללי השאילתות והדוחות וטכנולוגיית כלי הניתוח הרב-מימדי, מבוססים על תהליכי אימות. הם מאפשרים למשתמש לחולל בקלות שאילתות מורכבות ולקבל במהירות יחסית את התוצאות. כלים אלה אינם מנחים את המשתמש בדרך כלשהי להצגת השאילתות ומה המסקנות המסתתרות בתוצאה המתקבלת. כלי הניתוח הרב-מימדיים מאפשרים את הצגת הנתונים בצורה מתוחכמת, ומתבססים במידה רבה על חישובים שונים הנעשים מראש. כלי ניתוח סטטיסטיים מאפשרים את הניתוח בין מספר משתנים, ומאפשרים להגדיר את המובהקות הסטטיסטית ביניהם.

❖ **תהליכים אנליטיים מבוססי גילוי (Discovery Based Analysis):** העיקרון העומד בבסיס שיטה זו שונה - לא הוכחת הנחות, אלא גילוין. בתפיסה זו המשתמש אינו מניח הנחות כלשהן, אלא מאפשר לכלים אוטומטיים (המבוססים על אלגוריתמים מתוחכמים) לפעול מול הנתונים ולחשוף את התבניות המסתתרות בהם. במילים אחרות, על המשתמש לגלות את היהלומים המסתתרים במעמקי הנתונים. תפיסה זו נתמכת על ידי טכנולוגיה חדשה בשם "כריית נתונים". תהליכי **כריית נתונים** מסוגלים לבחון מספר רב של משתנים בו-זמנית ולחשוף יחסים שונים ביניהם, להצביע על קשרים מובהקים או על חריגים מיוחדים. ניתן לומר שכריית הנתונים מתבססת על הנתונים ומנסה לזהות מתוכם את היחסים, ההקבצות, התבניות וההקשרים. תהליכים אלה פועלים, בדרך כלל, באופן טוב יותר מול בסיסי נתונים גדולים, המכילים מספר רב של תצפיות או עובדות.

כדי להדגיש את ההבדלים בין שני סוגי התהליכים האנליטיים האלה נשתמש בדוגמה נוספת, הפעם מתחום השיווק והדיוור הישיר. חברה מבקשת לצאת במבצע שיווק למוצר חדש ועליה להחליט למי מלקוחותיה לשלוח מידע על מוצר חדש. המטרה היא להגיע ללקוחות שקיימת הסבירות הגבוהה ביותר שיהיו מעוניינים לרכוש את המוצר. אם תהליך החלטה זה יתבסס על מודל אימות, מנתח המידע יבנה לעצמו מודל כלשהו שיניח מה הן התכונות הרלוונטיות ביותר של הלקוחות, כמו למשל אזור מגורים, רמת הכנסות, תבנית הרכישות הקודמות שלו וכדומה. בהתבסס על מודל זה, הוא יתחיל להפעיל סדרת שאילתות על הנתונים, או לחילופין יתחיל להתבונן על הנתונים מזוויות שונות. הוא יבחן את התוצאות, ישנה את השאילתות או את נקודות המבט ויחזור על התהליך עד שיגיע אל רשימת הלקוחות הרצויה.

לא ניתן לומר שבתהליך זה נוצר ידע חדש, אלא התבצע תהליך אימות ההנחות מול הנתונים. כעת, נתבונן באותה בעיה של הגדרת אוכלוסיית הלקוחות לדיוור ישיר, אלא שהפעם נתבסס על מודל גילוי. בשיטה זו מפעיל מנתח המידע תהליכים אנליטיים שונים לסיווג הלקוחות לקבוצות שונות, מבלי להניח מראש מהי קבוצת הלקוחות המתאימה ביותר למבצע. לאחר שיאתר את ההקבצות השונות האפשריות של הלקוחות, הוא יחליט מהי הקבוצה המתאימה ביותר למבצע החדש. במקרה זה בוצע החיפוש אחר ההקבצות, ללא הנחה כלשהי מראש, ולמעשה היה כאן תהליך של גילוי ההקבצות המתאימות לאוכלוסיית הלקוחות. כמובן שניתן לגשת לבעיה זו בשיטות אנליטיות נוספות, כמו למשל הבנת מאפייניהם של לקוחות הקונים מוצרים דומים ובניית עץ החלטות שיופעל על כלל אוכלוסיית הלקוחות.

הדרך הנכונה להתייחס לשני סוגי התהליכים האנליטיים האלה היא כאל תהליכים משלימים. הכלים לכריית נתונים אינם מחליפים את מחוללי השאילתות ואת כלי הניתוח הרב-מימדי השונים, אלא מהווים נדבך נוסף ומשלים לכלים אלה. השלמה זו נועדה לאפשר מיצוי נרחב יותר של הידע מבסיס הנתונים.

חלק מהידע יכול להיווצר באמצעות מחוללי שאילתות וכלי הניתוח הרב-מימדי, אולם קיים רובד נוסף של ידע, המסתתר בבסיס הנתונים שאינו בר-גילוי באמצעות שאילתות SQL, מתוחכמות ככל שתהיינה, או הצגה רב-מימדית של המידע. לדוגמה, לא ניתן לנסח שאילתה שתבנה ותציג מהם פלחי הלקוחות, או מהן הסיבות למגמות הקניה בשנתיים

האחרונות. ניתן כמובן לנסות ולענות על שאלות כאלו באמצעות שאילתות רגילות, אולם זהו תהליך מייגע הדורש הבנה מעמיקה מאוד של הפרמטרים השונים. הוא דורש להניח מספר רב של הנחות ולהתחיל בתהליך של ניתוח, שברובו מבוסס על **ניסוי וטעייה** (Trial and Error). תהליך כזה יכול להימשך זמן רב ואין כל ביטחון שאכן נמצה בעזרתו את הידע המצוי בבסיס הנתונים.

סיווג הטכנולוגיות התומכות בניתוח נתונים

במשך השנים פותחו כלים וטכנולוגיות שנועדו לאפשר את ביצוע התהליכים האנליטיים של עיבוד וניתוח נתונים ומידע. ניתן לחלק כלים אלה לשלוש קבוצות עיקריות:

- ❖ מחוללי שאילתות ודוחות.
- ❖ כלים לניתוח רב-מימדי.
- ❖ כלים לכריית נתונים.

לניתוח רב-מימדי וכריית נתונים הוקדשו שני פרקים נפרדים בספר זה, גם בשל חשיבות הנושא, וגם בשל המורכבות היחסית שלהם.

מחוללי שאילתות ודוחות (Query and Report Generators)

המחוללים קיימים כבר שנים רבות והחלו להתפתח הרבה לפני הופעת תפיסת מחסן הנתונים. הצורך בהפקת דוחות ובבניית שאילתות מול בסיסי נתונים קיים גם במערכות התפעוליות, והחל להתפתח לאור צרכים אלה. עם השנים התפתחו המחוללים והפכו למשוכללים יותר, רבי עוצמה וידידותיים למשתמש. הם קיבלו דחיפה חזקה מאוד עם הופעת בסיסי הנתונים הטבלאיים ושפת SQL. שפת SQL היא בעלת יכולות רבות לתחקור ולהפקת מידע מבסיסי נתונים והפכה לסטנדרטית. על כן, יצרן שבנה כלי שתמך בשפת גישה זו לנתונים, נהנה מיכולת גישה למיגוון בסיסי נתונים טבלאיים של יצרנים שונים. עם הופעת טכנולוגיית ODBC, שפותחה וקודמה על ידי חברת מיקרוסופט, הפכו בסיס הנתונים הטבלאיים לפתוחים ונוחים מאוד לגישה ממיגוון כלים של צד שלישי.

כלי התחקור הפכו במשך הזמן לידידותיים מאוד, ולמעשה הם מאפשרים למשתמש שלא למד את שפת SQL לבנות שאילתות מורכבות למדי. כל זאת באמצעות ממשק גרפי נוח ושימוש בעכבר המשמש להצבעה על העמודות לשליפה, באמצעות הגדרה נוחה של הקריטריונים לשליפה, הגדרת סדר מיון של התוצאה וכדומה. עם הזמן התווספו לכלים אלה תכונות שמטרתן להקל על המשתמש את השימוש בהם, וביניהן:

- ❖ יכולת לפעול מול טבלאות מדומות (View). הטבלאות מוכנות מראש ומסתירות את מבנה בסיס הנתונים, את פעולות הצירוף (Join) שנעשות בין טבלאות ומכילות עמודות מדומות שנובעות מחישוב כלשהו.

- ❖ יכולת להגדיר מחדש את כל שמות העמודות והטבלאות. כך שהמשתמש יכול לעבוד עם בסיס הנתונים באמצעות מונחים הלקוחים מעולמו, ולא במונחים טכניים שנקבעו על ידי מנהל בסיס הנתונים.
- ❖ יכולת ניהול ספריית שאילתות פרטית. כך יכול המשתמש לשמור את השאילתות שערך, לשלוף אותן בעת הצורך ולהפעילן באמצעות פרמטרים שונים.
- ❖ שילוב גרפיקה עסקית וצבעים בפלט. המשתמש יכול לבחור מתוך מיגוון של סוגי גרפים שונים.
- ❖ ניהול הרשאות מתקדם מאפשר להגדיר למי מותר לעשות מה, פרופילים לניצול משאבי מחשוב למשתמשים שונים וכדומה.
- ❖ משלוח התוצאה למסך או מדפסת, על פי הגדרת המשתמש, העברת התוצאה בדואר אלקטרוני פנימי או העברה למבנה מתאים לטכנולוגיית אינטרנט.
- ❖ הפעלה אוטומטית (Scheduler) של השאילתות על פי קריטריונים שונים, כגון זמן (כל יום שישי), אירוע (למשל, כאשר סך כל המכירות בסניף מסוים ירדו מתחת לסף מסוים) וכדומה.

תכונות אלו, אשר הולכות ומשתכללות עם הזמן, הפכו כלים אלה לחלק בלתי נפרד מסביבת מחסן הנתונים. הכלים לתחקור הנתונים הם הרבה יותר מאשר מחולל שאילתות, ולכן מקובל לקרוא להם **סביבה מנוהלת לשאילתות** (Managed Query Environment). סביבה זו מספקת שירותים רבים, הן למשתמש הקצה והן למנהלי המערכת. בין הכלים המובילים בקבוצת מחוללי השאילתות נוכל למנות את Business Objects של חברת Business Objects SA, Discoverer/2000 של חברת Oracle Corp, Impromptu של חברת Cognos, Brio Enterprise של חברת Brio Technologies ורבים נוספים. מבין מחוללי הדוחות המובילים נוכל למנות את Crystal Reports של חברת Seagate, IQ/Vision של חברת IQ Software, R&R, ProReports, SQR ואחרים.

כלי ניתוח רב-מימדיים (OLAP - On Line Analytical Processing)

טכנולוגיית OLAP היא טכנולוגיה ייעודית, שנועדה לאפשר ניתוחי מידע מורכבים מבחינת המחשב, אבל פשוטים יחסית מנקודת מבטו של המשתמש. שיטת ניתוח זו אינה דורשת מהמשתמש להכיר את מורכבויות הניווט בבסיס הנתונים. מוצרי OLAP מארגנים את המידע במבנה קוביה רב-מימדית בה כל פיאה מייצגת מימד עסקי בעל משמעות: מוצרים, לקוחות, סניפים, זמן וכדומה. במפגש של כל אחד ממימדים אלה נשמרים הנתונים, או **העובדות** כפי שמקובל לכנותם, כמו למשל "מה היו המכירות של מוצר מסוים, בסניף מסוים, ללקוח מסוים בשבוע מסוים".

תפיסת ניתוח זו מבוססת על ההבנה שמקבלי החלטות מתחילים את תהליך ניתוח הנתונים מהתבוננות ברמה גבוהה יחסית של סיכומים, ובהדרגה מבקשים לקבל פרטים נוספים, בשיטת **מעלה-מטה** (Top-Down). בדרך כלל, ניתוח הנתונים אינו מתחיל

בהתבוננות ברמת המכירות של מוצר X בסניף Y בתאריך W, אלא בהתבוננות על סך המכירות לפי מוצר, לפי סניף ולפי תאריך. זו רמה **סיכומית** (Summarized), ולא רמה **מפורטת** (Detailed) של הנתונים. רמה סיכומית זו משמשת עוגן, שממנו מתחיל תהליך הניתוח המנסה לזהות מגמות וחריגים: אילו מוצרים נמכרו יותר, אילו סניפים מכרו פחות וכדומה. מתמונת-על זו ניתן להתחיל **בירידה לפרטים** (Drill Down) למוצר מסוים או לסניף מסוים.

למבנה הרב-מימדי יתרונות רבים על פני המבנה הטבלאי הפשוט, ולכן הוא הפך לפופולרי מאוד בקרב מנתחי המידע.

נקודת מבט טבלאית

מוצרים	יום בשבוע	סה"כ מכירות
מוצר א'	יום א'	2945
מוצר א'	יום ב'	3035
מוצר ב'	יום א'	1910
מוצר ב'	יום ב'	1910

נקודת מבט רב מימדית

	מוצר א'	מוצר ב'
יום א'	2945	1910
יום ב'	3035	2484

	יום א'	יום ב'
מוצר א'	2945	3035
מוצר ב'	1910	2484

תרשים 3.7: הצגה טבלאית מול הצגה רב-מימדית של נתונים.

כפי שניתן לראות מתרשים 3.7 הצגה רב-מימדית של הנתונים היא קומפקטית ונוחה יותר. התרשים גם מציג את אחת התכונות הבסיסיות של הצגה זו: יכולת **סיבוב המימדים**.

הניתוח הרב-מימדי מבוסס על אינטראקציה מיוחדת בין המשתמש לבין הנתונים. אינטראקציה זו ניסוח שאילתות במבנה מיוחד, על היכולת לנווט במבנה מורכב של נתונים באופן פשוט, על היכולת להתבונן בנתונים בחתכים שונים וגמישים, על היכולת לרדת לפרטים בהתאם לתוצאות השאילתה, ועל היכולת לעלות לרמות סיכום גבוהות יותר. הרב-מימדיות מתבטאת ביכולת להציג את המידע בצורה מיוחדת וגמישה, שמזכירה את מבנה הקוביה הרב-מימדית שניתן לסובב אותה בכיוונים שונים. לתהליכי הניתוח הרב-מימדי מוקדש פרק נפרד בהמשך, ולכן לא נרחיב כאן.

כלי כריית נתונים (Data Mining)

כריית נתונים היא אחד הנושאים המעסיקים מאוד את אנשי טכנולוגיית המידע בשנים האחרונות. אין כנס שמכבד את עצמו ועוסק בטכנולוגיות מידע מודרניות ובמחסני נתונים, בו לא מוקדשות מספר הרצאות לנושא זה. כמות המאמרים והספרים העוסקים בטכנולוגיה זו הולכת וגדלה וכמעט ורוב הספקים בתחום טכנולוגיות המידע עוסקים בפיתוח ובשיווק מוצרים לכריית נתונים. טכנולוגיה זו מבוססת על רשתות עצביות, אלגוריתמים גנטיים, עצי החלטות, אינטליגנציה מלאכותית ומערכות לומדות. שמות אלה גרמו לכך שבעיני חלק מהאנשים נשמעת טכנולוגיה זו כמו מדע בדיוני, ולא כמו טכנולוגיה בשלה שיכולה לסייע לארגון שלהם כבר היום. טכנולוגיה זו אינה מדע בדיוני, אלא הבשלה ומימוש של נושאים מתקדמים שנחקרו במשך שנים בתחום מדעי המחשב.

למעשה, טכנולוגיית כריית הנתונים היא אכן טכנולוגיה חדשה, המבוססת על אלגוריתמים שונים, שבמצבים מסוימים הוכיחו יכולת גילוי טובה מאוד של ידע חדש מתוך בסיסי נתונים גדולים. תחום כריית הנתונים מורכב ממספר ענפים שונים. כל ענף מבוסס על סוג מסוים של אלגוריתמים ומתאים לסוג מסוים של מצבים. לא ניתן להשתמש בכלי אחד לכל היישומים והמצבים. שימוש בכלים אלה דורש רמת מיומנות גבוהה יותר מאשר בכלים האחרים. כעיקרון, מקובל לחלק תחום זה למספר קטגוריות שונות של טכנולוגיות וכלים:

- ❖ **כלים סטטיסטיים** (Statistical Analysis): מקובל להתייחס אל כלים אלה כאל כלים בסיסיים לכריית נתונים. הכלים מאפשרים ניתוח נתונים מסוגים שונים: שונות, סטיות תקן, רגרסיות לינאריות, בדיקת מובהקות סטטיסטית, ניתוח סדרות עיתיות וכדומה. רוב הכלים מאפשרים הצגה גרפית של הנתונים. השימוש בכלים אלה דורש מיומנות מצד המשתמשים והבנה בתיאוריה הסטטיסטית. בין הכלים המובילים בקטגוריה זו ניתן למנות את SAS של חברת SAS Institute, SPSS של חברת SPSS Inc, Cornerstone של חברת BBN Software Products, Data של חברת TreeAge Software ואחרים.
- ❖ **כלי הצגה** (Visualization Tools): כלים אלה מאפשרים הצגה מתוחכמת של נתונים, כמו הצגת פיזור נתונים, הצגה תלת-מימדית, שילוב מידע גיאוגרפי עם מידע נומרי, אנימציה של הצגת הנתונים על ציר הזמן וכדומה. כלים אלה היו בשימוש בעיקר בקהיליית החוקרים והמהנדסים, ורק לאחרונה חדרו לשימושים עסקיים. הצגת הנתונים מאפשרת למשתמש מתוחכם להתבונן בהם בדרכים חדשות, ולגלות תבניות בלתי ידועות. לדוגמה, הם מאפשרים לגלות שרוב רוכשי מוצר מסוים באזור מסוים, הם גברים בגילאים מסוימים.
- ❖ **כלים לגילוי יחסים ועצי החלטות** (Rule and Decision-Tree Discovery): כלים אלה הם הנפוצים ביותר מבין כלי כריית הנתונים ומאפשרים גילוי חוקים ויחסים בלתי ידועים המסתתרים בבסיס הנתונים. חלק מהצגת התוצאה כוללת הצגת חוקים, כגון "קיימת הסתברות של 83% שלקוח הרוכש A וגם רוכש B ירכוש גם C". חלק מחוקים אלה מוצגים בצורה של עצי החלטות. הכלים בונים את הידע מתוך סריקת בסיסי נתונים היסטוריים גדולים, מרכיבים את החוקים ואת היחסים ומאפשרים חיזוי של מצבים חדשים.
- ❖ **רשתות עצביות** (Neural Networks): כלים אלה מבוססים על מודל של רשת עצבית. זהו חיקוי מסוים ומוגבל של צורת פעולת המוח האנושי, עד כמה שאנו מסוגלים להבין נושא מורכב זה. כלים אלה טובים מאוד בפילוח אוכלוסיות ומתאימים למצבי חיזוי, במצבים בהם הנתונים לא נקיים ומכילים רעש רב. הבעיה העיקרית של כלים אלה בכך שהם מתנהגים כמו "קופסה שחורה" ולכן קשה לעיתים להבין מדוע התקבלה התוצאה שהתקבלה.

- **לוגיקה עמומה** (Fuzzy Logic): אוסף חדש יחסית של כלים המבוססים על תורת הלוגיקה העמומה, להבדיל מהלוגיקה הרגילה של אמת ושקר. אלגוריתמים אלה מסוגלים לבצע חיפושים מתוחכמים בנתונים, ולמצוא מצבים קרובים למצב מסוים, ולא רק מצבים התואמים באופן מלא למצב מסוים.

- **מערכות מבוססות ידע** (Knowledge Based Systems): אוסף חדש יחסית של כלים, המבוסס על תהליך של הנדסת ידע בו אוספים את הידע ממומחה אנושי, מנסחים אותו עבור המערכת ומאפשרים לה, על בסיס ידע זה, לאתר בנתונים מצבים מסוגים שונים. כלים אלה נדירים מכיון ששימוש בהם מחייב מעורבות של מומחים, ובתהליכים מורכבים לניסוח הידע באופן פורמלי.

כיום קיימים עשרות מוצרים שונים, חלקם מתאים לנושא מסוים ומבוסס על טכניקה אחת, בעוד שאחרים תומכים במספר טכניקות ולכן מתאימים לסוגי יישומים שונים. בין המוצרים המובילים לכריית נתונים נזכיר את Data Mind Professional של חברת DataMind Inc, ‏Intelligent Miner של חברת IBM, ‏Darwin של חברת Thinking Machines, ‏MineSet של חברת Silicon Graphics ואחרים.

כאמור, לכריית נתונים מוקדש פרק נפרד בספר, ולכן לא נרחיב את הדיון בנושא זה כעת.

סיווג מרכיבי המידע

ניתן להסתכל על המידע בשלושה מישורים שונים:

- **תוכן המידע** (Information Content) - תוכן המידע הוא המרכיב הבסיסי ביותר שלו. זוהי **משמעות הנתונים**. התוכן נקבע על ידי הנושא בו עוסק המידע, כמו למשל מידע לגבי מכירות מוצרים, מידע רפואי אודות חולים, מידע על עובדים, מידע לגבי תקלות בקו הייצור וכדומה. מקובל להבחין בין סוגי מידע שונים וביניהם מידע נומרי, מידע טקסטואלי, מידע מרחבי או גיאוגרפי, מידע קולי וכדומה.

- **מבנה המידע** (Information Structure) - המבנה עוסק בצורה בה אורזים ומנהלים את המידע. קיימים מספר מבנים שונים לאריזת מידע עבור תהליכי קבלת החלטות. הצורה הפשוטה ביותר היא במבנה טבלאי, כפי שמנהל כל בסיס נתונים יחסי או בצורה מורכבת יותר של מבנה רב-מימדי, שהוא אחד ממבני המידע הנפוצים ביותר לניתוח מידע.

- **מנגנון אספקת המידע** (Information Delivery Mechanism) - אספקת המידע עוסקת במנגנונים המספקים את המידע אל שולחן המשתמש. סוגים שונים של משתמשים יהיו זקוקים לטכנולוגיות אנליטיות שונות לאספקת המידע. בסעיף המפרט את הטכנולוגיות השונות הזכרנו מנגנוני אספקה שונים, מהפשוטים ביותר, כגון מחוללי הדוחות והשאילתות והכלים לגרפיקה עסקית, ועד למנגנוני אספקה מורכבים המבוססים על כלי ניתוח מידע רב-מימדיים וכריית נתונים.

סיווג המשימות האנליטיות

בארגון קיימים מספר סוגים של **עובדי ידע**, העוסקים בניתוח מידע ובקבלת החלטות, או כפי שהם נקראים לעיתים: **עובדי צווארון לבן**. עובדים אלה מבצעים משימות שונות הקשורות בניתוח מידע. נסווג את המשימות שעובדים אלה מבצעים לשלוש קטגוריות:

- ❖ **מעקב ודיווח** (Monitoring and Reporting): משימה זו היא הפשוטה ביותר המבוצעת על ידי המשתמשים. מטרתה לעקוב באופן שוטף אחר פרמטרים שונים של הארגון ולאתר בעיות, במידה והן מתרחשות. משימות אלו מאופיינות בכך שדרישות המידע לביצוען ידועות היטב. המשימה חוזרת על עצמה מדי תקופה ומבוססת במידה רבה על איתור מצבים חריגים. הכלים המתאימים ביותר לביצוע משימות מסוג זה הם מחוללי הדוחות והשאילתות וכלים פשוטים לגרפיקה עסקית. מקובל לקרוא לכלים אלה גם בשם הכולל **סביבות מנוהלות לשאילתות** (Managed Query Environments). הן מאפשרות ניסוח מהיר וקל של הדרישות למידע, תוך שימוש במונחים עסקיים (לא טכניים) של בסיס הנתונים, אספקת נקודות מבט פשוטות (Views) על בסיס נתונים מורכב, שמירת השאילתות והפעלתן בעת הצורך, מנגנוני הרשאות חזקים וכדומה. שפת SQL מתאימה מאוד לביצוע משימות אלו. לכן, בסיסי הנתונים היחסיים הפכו מאוד פופולריים לסוג זה של ניתוחי מידע. לאיתור מצבים חריגים ניתן גם להשתמש בכלים נוספים, כגון Triggers, המופעלים באופן אוטומטי כאשר מתרחש מצב חריג, ומפעילים תהליכי דיווח והתראה מתאימים.
- ❖ **ניתוח ואבחון** (Analysis and Diagnosis): משימה זו עוסקת במיגוון פעולות ניתוח מידע העוסקות בגילוי בעיות ומציאת פתרונות. לסוג זה של משימות מתאימים מאוד כלי הניתוח הרב-מימדי (Multi Dimensional Analysis) וכלים שונים לסריקת בסיס הנתונים (Data Base Browsing). כלי הניתוח הרב-מימדיים מאפשרים הצגה נוחה של המידע בחתכים גמישים ביותר, אפשרויות לקבל פירוט נוסף, קבלת מידע ברמה סיכומית יותר ועוד. ההבדל בין כלי השאילתות לבין כלי הסריקה דומה להבדל בין מצב בו אנו יודעים מה אנו מחפשים למצב בו אין אנו יודעים מה אנו מחפשים. כלי סריקה של בסיס הנתונים צריך להתחיל סריקה בכל נקודה ולסיים אותה בכל נקודה. הוא צריך לתמוך בתהליכים איטרטיביים, בהם תוצאה של סריקה מסוימת מוזנת כקלט לסריקה חדשה וכדומה. חלק מהכלים משתמשים לסריקת בסיס הנתונים בממשק של גיליון אלקטרוני, בעוד שאחרים משתמשים בממשק רב-מימדי.
- ❖ **סימולציה ותכנון** (Simulation and Planning): משימה זו היא המורכבת ביותר, שלביצועה דרושה רמת מיומנות גבוהה. הכלים התומכים במשימות אלו צריכים לאפשר בחינת תרחישים עסקיים שונים, כגון ניתוחי What If שבהם בוחנים מה יקרה אם ישתנו חלק מכללי המשחק, חיזוי של מצבים עתידיים, גילוי יחסים ותבניות בלתי ידועות עד כה מהנתונים וכו'. כלים אלה מחייבים גם יכולת לעדכן את בסיס הנתונים כתוצאה משינוי הפרמטרים (למשל, להציג את מצב רווחיות החברה אם הריבית במשק תעלה). בקטגוריית הכלים העונים למשימות אלו נמצא כלי כריית נתונים, כלים להצגת נתונים, מודלים מתמטיים, מודלים סטטיסטיים לחיזוי ועוד.

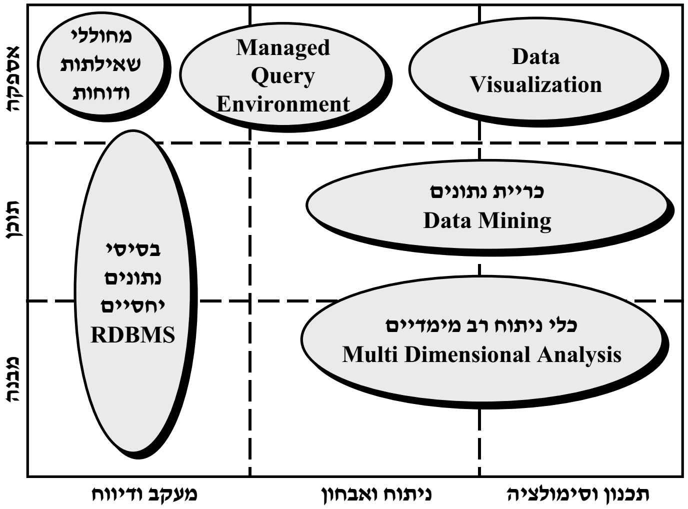
סיווג היישומים האנליטיים לפי הדרג הארגוני

דרך נוספת להתבונן בתהליכים האנליטיים היא דרך הרמה הארגונית המבצעת את התהליכים. מקובל להבחין בסוגים שונים של יישומים אנליטיים, בהתאם לסוג ההחלטות הנתמכות על ידם: יישומיים תפעוליים, יישומים טקטיים ויישומים אסטרטגיים. כפי שנראה, קיימת מידה רבה של הקבלה בין סיווג התהליכים לפי הרמה הארגונית לבין סיווג משימות הניתוח, כפי שהוצג בסעיף הקודם. נסקור בקצרה כל אחד מסוגי יישומים אלה.

- **יישומים אנליטיים תפעוליים** - יישומים אלה משרתים את הארגון בתהליכים השוטפים של קבלת ההחלטות. בדרך כלל הם עוסקים במעקב שוטף אחר ביצועי הארגון, בשינויים במגמות ובאיתור מצבים חריגים. מכיון שיישומים אלה מבוססים על הנתונים השוטפים של הארגון, הם מבוססים למעשה על ניתוחים פרטניים של נפחי נתונים גדולים. העומק ההיסטורי עבור יישומים אלה הוא ימים ועד מספר חודשים. היישומים נתמכים היטב על ידי מחוללי שאילתות ודוחות הפועלים מול בסיסי נתונים טבלאיים, המסוגלים לנהל נפחי נתונים גדולים מאוד.
- **יישומים אנליטיים טקטיים** - יישומים אלה עוסקים בניתוחי שוק ובתכנון טקטי, כגון ניתוח רווחיות המוצרים, רווחיות ברמת לקוח, פילוח אוכלוסיית הלקוחות. בדרך כלל מתבצעים ניתוחים אלה ברמה סיכומית יותר, ולא ברמה הפרטנית של הנתונים. לכן, כלי ניתוח רב-מימדיים מתאימים מאוד לסוג כזה של יישומים. העומק ההיסטורי של יישומים אלה נמדד בחודשים עד שנה.
- **יישומים אנליטיים אסטרטגיים** - יישומים אלה עוסקים בניתוחים מורכבים של הנתונים כדי לאתר מוצרים חדשים שניתן להציע, הבנה טובה יותר של צרכי הלקוחות, ניתוחי השוק והמתחרים, החלטה על שווקים חדשים, ניתוח תרחישים עתידיים ומה השפעתם על הארגון וכדומה. הכלים המתאימים ביותר בתחום זה הם כלים לכריית נתונים וכלים סטטיסטיים. העומק ההיסטורי הנדרש לניתוחים אלה הוא בדרך כלל הרב ביותר, ויכול להגיע גם למספר שנים.

מפת המרחב האנליטי (Analytical Space)

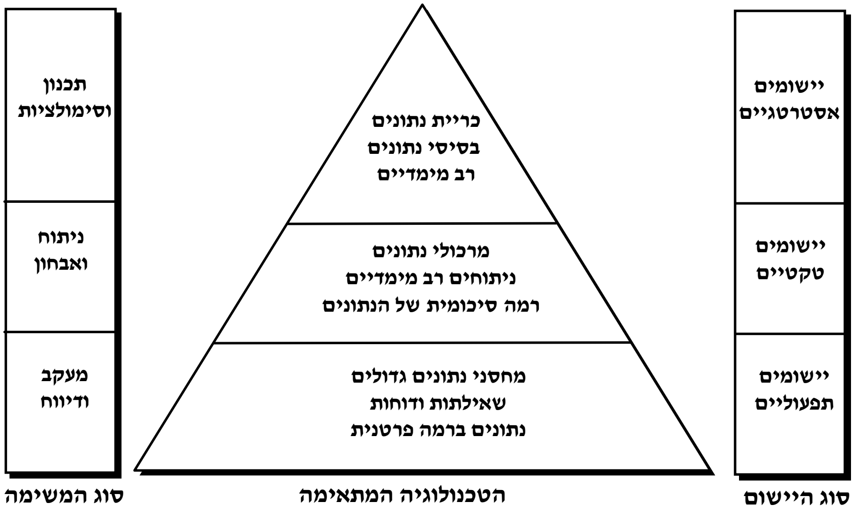
כעת, לאחר שסקרנו את סוגי התהליכים האנליטיים, את סוגי המשימות האנליטיות השונות, את הדרג הארגוני המבצע את המשימות האנליטיות, ואת הטכנולוגיות האנליטיות השונות, נוכל להציג את המרחב האנליטי. עלינו להבחין בין שלושה מישורי מידע - תוכן, מבנה ומנגנון אספקה - לבין שלוש קטגוריות עיקריות של משימות המבוצעות על ידי עובדי הידע בארגון - מעקב ודיווח, ניתוח ואבחון, סימולציה ותכנון. על בסיס ההבחנה הזו נוכל לצייר מפה המציגה את כל המרחב האנליטי ואת הטכנולוגיות המתאימות ביותר לביצוע משימות הניתוח השונות.



תרשים 3.8: מפת המרחב האנליטי לניתוח מידע.

מהמפה המופיעה בתרשים 3.8 צריך להיות ברור לקורא שאין כלי ניתוח אנליטי אחד ויחיד העונה **לכל** סוגי המשימות ומתאים **לכל** המשתמשים. סביר להניח שקביעה זו תהיה נכונה גם לשנים הבאות. לכן חשוב שנבין שבתפיסת מחסן הנתונים יש מקום למספר כלים שונים, המיועדים למשתמשים שונים ולמשימות ניתוח מידע שונות.

דרך נוספת להתבונן במרחב האנליטי היא לבחון את סוגי היישומים ואת טכנולוגיות של מחסן נתונים המתאימות להם ביותר.



תרשים 3.9: סוגי הטכנולוגיות לפי סוגי היישומים.

סיכום

מטרת פרק זה היתה לסקור את המרחב האנליטי של ניתוחי המידע בסביבת מחסן הנתונים ואת תהליך הפיכת הנתונים להחלטות עסקיות. המרחב האנליטי עתיר בסוגי מידע שונים, בסוגי משימות ניתוח שונות, ביישומים מסוגים שונים וכמובן - בטכנולוגיות שונות. אין, ולא תהיה טכנולוגיה אחת המתאימה **לכל** העושר של המרחב האנליטי. הארגון צריך לבחון במסגרת תהליך הקמת סביבת מחסן נתונים, איזה יישומים ומשימות יבוצעו. בהתאם למסקנות שיוסקו יש לבחור בטכנולוגיה ובכלים המתאימים ביותר.

כל ארגון חייב להגדיר מהן המשימות האנליטיות שברצונו לבצע בסביבת מחסן הנתונים, ולאתר את הכלים המתאימים לביצוע עיבודים אלה.

פרק 4: ניתוח מידע רב-מימדי (Multi Dimensional Analysis)

מבוא

פרק זה עוסק באחת מהשיטות האנליטיות הנפוצות ביותר בסביבת מחסן הנתונים - **ניתוח רב-מימדי של מידע**. הניתוח הרב-מימדי מבוסס על תפיסה המאפשרת למשתמשים ולמקבלי ההחלטות לראות את "התמונה הגדולה" של המידע (הכנסות, הוצאות, שווקים, מוצרים וכדומה), תוך מתן אפשרות להתבונן ולנתח אותו במיגוון רחב של נקודות מבט וחתכים שונים. בשל יכולות אלו זכה הניתוח הרב-מימדי לפופולריות עצומה בקרב אוכלוסיית המנהלים ומנתחי המידע. הניתוח הרב-מימדי עוסק בצורת אינטראקציה מיוחדת בין המשתמש לבין הנתונים, אינטראקציה המבוססת על שאילתות במבנה מיוחד, על יכולת לנווט בפשטות במבנה מורכב של נתונים, על יכולת להתבונן בנתונים בחתכים שונים וגמישים, על יכולת להצגת הנתונים ברמות סיכומיות ומשם לרדת לרמות מפורטות יותר, בהתאם לתוצאות השאילתה, או לחילופין על יכולת לעלות מרמת פירוט מסוימת לרמות סיכומיות גבוהות יותר. הרב-מימדיות מתבטאת ביכולת להציג את המידע בצורה מיוחדת וגמישה, צורה המזכירה מבנה קוביה רב-מימדית, אותה ניתן לסובב בכיוונים שונים. כל מימד בקוביה מייצג מימד עסקי כלשהו - לקוחות, מוצרים, סניפים וכדומה. הגמישות בגישה לנתונים היא אחת מיכולות היסוד של גישה זו, בשל העובדה שתהליך הניתוח מתחיל, בדרך כלל, מנקודה מסוימת ומשם מתחילים לנווט במרחב הנתונים בצורות שונות. הניווט מתבצע לכיוונים שונים ובלתי ידועים מראש, כאלה הנקבעים על ידי המידע המתגלה, תוך כדי תהליך הניתוח. הטכנולוגיה והכלים התומכים בשיטת הניתוח הרב-מימדי זכו לשם הכולל של **כלי OLAP**.

מושגי היסוד בניתוח רב-מימדי - עובדות ומימדים

הניתוח הרב-מימדי מבוסס על העיקרון שניתן להתבונן בנתונים מנקודת מבט של המימדים העסקיים המשמשים את הארגון בניהול פעולותיו. לדוגמה, ניתן להתבונן על נתוני המכירות של הארגון, על פי מימדים כגון מוצרים, סניפים, לקוחות או זמן, או להתבונן בהתנהגות מנויים של חברת טלפון על פי מימדים כגון לקוחות, מספרי טלפון קוראים, מספרי טלפון נקראים, סוג השיחה שבוצעה, רצועת זמן בה השיחה בוצעה, או להתבונן בפעילותה של חברת ביטוח על פי מימדים כגון לקוחות, סוכנים, סוגי פוליסות ביטוח, זמן וכדומה. הבה נתחיל לפתח רעיון זה באופן שיטתי יותר.

בעיקרון, ניתן לחלק את יישויות המידע המנוהלות במערכות התפעוליות ליישויות בעלות אופי סטטי ויישויות בעלות אופי דינמי. היישויות הסטטיות הן יישויות שקצב העדכונים שלהן נמוך יחסית, כגון לקוחות, מוצרים, סניפים, פוליסות ביטוח, חשבון הבנק וכדומה. היישויות הדינמיות הן יישויות המייצגות את האירועים השוטפים בארגון, כגון מכירת מוצר, ביצוע שיחת טלפון, קבלת תביעה בגין פוליסת ביטוח, תנועה בחשבון עו״ש וכדומה. בשל האופי הדינמי שלהן, מקובל גם לקרוא לאירועים אלה בשם **תנועות עסקיות** (Business Transactions). בין כל סוגי היישויות מתקיים מיגוון רחב של קשרים, כגון שיחות הטלפון של לקוח מסוים, תנועות מלאי של מוצר מסוים, תנועות בחשבון של לקוח מסוים וכדומה. כפי שנראה מייד, מבחינת מערכות תפעוליות אין אבחנה מיוחדת בין שני סוגי יישויות אלו, אולם מבחינת תהליך ניתוח הנתונים, האבחנה ביניהם קיימת והיא בעלת חשיבות.

לכל יישות, בין אם היא סטטית או דינמית, יש מספר **מאפיינים** (Attributes). לדוגמה, המאפיינים של היישות **לקוח** הם מספר לקוח, שם לקוח, כתובת לקוח, תאריך פתיחת רשומת הלקוח וכדומה. המאפיינים של היישות **מכירה** הם מספר הזמנת לקוח, קוד המוצר שנמכר, כמות המכירה, תאריך המכירה, איזה סניף ביצע את המכירה ומה המחיר בו רכש הלקוח את המוצר. במערכת בנקאית יש ליישות **תנועה בחשבון** מאפיינים כגון מספר החשבון, הסניף בו בוצעה הפעולה, תאריך הפעולה, סכום הפעולה, האם זו הפקדה או משיכה וכדומה.

למשל, נתבונן בדוח אופייני של מכירות, המכיל שורה לכל מכירה שבוצעה.

מספר הזמנה	שם לקוח	מוצר	מבצע הנחות	צינור הפצה	תאריך	חנות	מחיר מוצר	כמות
45-2389	אייל הייפרמן	טלויזיה סוני דגם 123	20%	הפצה ישירה	12.02.98	נוה סביון	4,500	2
45-3512	רן כהן	טלויזיה סוני דגם 123	20%	סוכנות הפצה	15.02.98	רמת אפעל	4,500	1

תרשים 4.1: דוח מכירות.

כל שורה בדוח זה מייצגת אירוע של מכירה בודדת של מוצר מסוים שבוצעה בחנות מסוימת בתאריך כלשהו וכדומה. אירועים אלה נרשמים במערכות התפעוליות של הארגון. בדוגמה שלנו, מערכת הזמנות לקוח רושמת כל הזמנה שלקוח כלשהו מבצע.

למטרת ניתוח המידע וקבלת החלטות, הרמה הפרטנית המופיעה בדוח זה מעניינת פחות ובדרך כלל נדרשת רמה גבוהה יותר, למשל סיכום יומי של המכירות שבוצעו לכל מוצר, בכל חנות. לקבלת דוח סיכומי כזה נוכל לסכם את התנועות העסקיות הבסיסיות, ונקבל את הדוח הבא.

מוצר	מבצע הנחות	צינור הפצה	תאריך	חנות	סך מכירות	כמות
טלויזיה סוני דגם 123	20%	הפצה ישירה	12.02.98	מפרץ חיפה	90,000	20
טלויזיה סוני דגם 123	20%	סוכנות הפצה	12.02.98	נוה סביון	45,000	10

תרשים 4.2: דוח סך מכירות יומי.

דוח זה הוא דוח תוצאתי הנובע מסיכום כל תנועות המכירה שבוצעו ביום מסוים. מתוך הדוח ניתן להבין שסך המכירות ליום 12.02.98 של טלוויזיות מסוג מסוים בחנות מפרץ חיפה היו 90,000 ש״ח. מכיון שזהו דוח סיכומי, לא מופיע בו שם הלקוח ומספר ההזמנה הבודדת. דוח סיכומי כזה יכול לשמש לקבלת החלטות, אם למשל נשווה את המכירות בין ימים שונים, או חנויות שונות או מוצרים שונים, ונזהה מגמה מסוימת או חריגה כלשהי. למשל, שמכירות של מוצר מסוים בחנות מסוימת גבוהות ב- 40% מהממוצע, או שסך המכירות בימי ה׳ תמיד גבוה ב- 15% מימי השבוע האחרים, וכדומה. בדרך כלל, מהווה גילוי מגמות או חריגות אלה את תחילת תהליך קבלת החלטות, כגון מבצעי עידוד מכירות בימים מסוימים למוצרים מסוימים, או תגבור צוות המכירות בחנויות מסוימות וכדומה. דוחות סיכומיים כאלה יאפשרו למנהלים להתחיל לשאול מיגוון שאלות מעניינות, כגון מהי החנות שמכרה הכי הרבה ממוצר מסוים, מהו התאריך בו מכרנו הכי הרבה, האם מבצע ההנחות אכן הגדיל את המכירות, מהם המוצרים הנמכרים ביותר וכדומה.

האופן הטבלאי של הצגת הנתונים, כפי שהוא מופיע בדוח, אינו נוח לצורך ניתוחי המידע. החיפוש אחר צורות הצגת נתונים נוחות יותר הביא להתפתחות תפיסת הניתוח הרב-מימדי. בדוח הסיכומי מסתתרים שני סוגים שונים של עמודות - עמודות המייצגות עובדות (Facts) ועמודות המייצגות מימדים עסקיים (Business Dimensions).

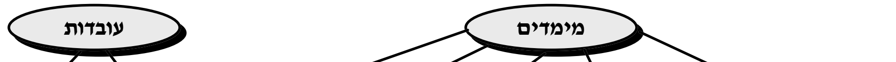
נתבונן לרגע בשתי העמודות האחרונות בטבלה, סך מכירות וכמות המכירות.

סך מכירות	כמות
90,000	20
45,000	10

תרשים 4.3: עמודות העובדות בתוך הדוח.

עמודות אלו מייצגות את תוצאת האירועים העסקיים, ולכן נקרא להן **עובדות**. השורה הראשונה מייצגת את העובדה שבוצעו מכירות בסך 90,000 ש״ח ונמכרו 20 יחידות, ואילו השורה השנייה מייצגת את העובדה שבוצעו מכירות בסך 45,000 ש״ח ונמכרו 10 יחידות. ניתן להגיע לנתונים המוצגים בעמודות אלה על ידי עיבוד כלשהו המתבצע על היישויות הדינמיות המנוהלות במערכת התפעולית. למטרות ניתוח אנו מתעניינים בעיקר בעמודות המייצגות עובדות נומריות, כמו סך המכירות בש״ח, סך כניסות פריט למלאי, משך דקות שיחה של לקוח וכדומה.

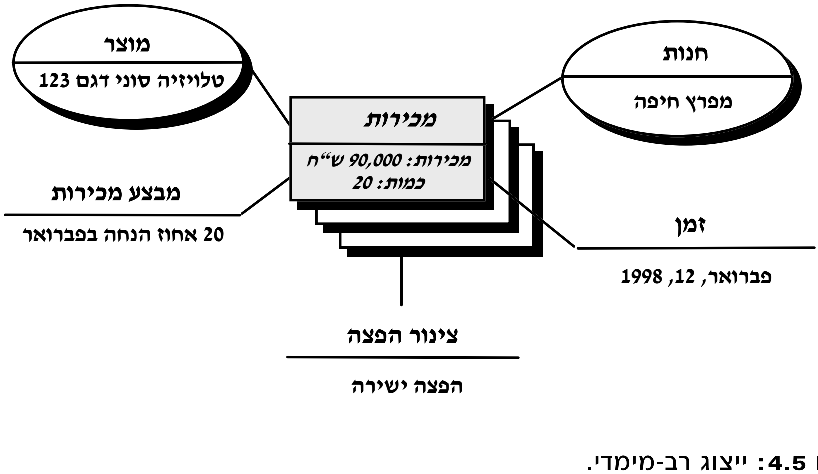
כמובן, שהעובדות המופיעות בתרשים 4.3 הן חסרות כל משמעות מבחינת המשתמש. כדי לתת לעובדות הבסיסיות המופיעות בדוח זה משמעות, עלינו להוסיף עמודות המספקות את המשמעות העסקית. לעמודות חדשות אלו נקרא **מימדיים עסקיים**, כגון מוצר, מבצע, צינור הפצה, חנות, זמן וכדומה. התרשים הבא מציג מחדש את הדוח שבתרשים 4.2, אבל הפעם - עם האבחנה בין העמודות המייצגות עובדות, לבין העמודות המייצגות מימדים עסקיים.



מוצר	מבצע הנחות	צינור הפצה	תאריך	חנות	סך מכירות	כמות
טלויזיה סוני דגם 123	20%	הפצה ישירה	12.02.98	מפרץ חיפה	90,000	20
טלויזיה סוני דגם 123	20%	סוכנות הפצה	12.02.98	נוה סביון	45,000	10

תרשים 4.4: דוח מכירות עם אבחנה בין עובדות ומימדים עסקיים.

במקום הייצוג השטוח והדו-מימדי המופיע בתרשים 4.4, ניתן להציג את הנתונים גם בצורה המחדדת את האבחנה בין העובדות לבין המימדים. במרכז נציג את העובדות ונקיף אותן במימדים עסקיים. מכיון שסביב עובדה מסוימת נמצא, בדרך כלל, מספר רב של מימדים, מקובל לקרוא למבנה המתקבל מבנה רב-מימדי.



תרשים 4.5: ייצוג רב-מימדי.

בכל נקודת מפגש של המימדים ב**מרחב הנתונים הרב-מימדי** (Multi Dimensional Data Space) הדמיוני שנוצר, מנוהלים הנתונים העובדתיים (Factual Data). למשל, סכום המכירות של מוצר מסוים, בתאריך מסוים, בחנות מסוימת. זהו הבסיס לצורת התבוננות מיוחדת בנתונים - המבט הרב-מימדי. ניתן להתייחס אל המבנה הרב-מימדי כאל הרחבה של מודל הגיליון האלקטרוני המטפל במבנה דו-מימדי.

לכל מימד ניתן לנהל מספר **מאפייני המימד** (Dimension Attributes). לדוגמה, למימד החנויות נוכל לנהל מאפיינים כגון שם החנות, כתובת החנות, גודל החנות במ"ר, ציון מיוחד אם החנות נמצאת בקניון או שהיא חנות עצמאית, ציון האם זו חנות חדשה או ותיקה, וכדומה. למימד הזמן נוכל לנהל מאפיינים כגון תאריך, יום בשבוע, ציון האם זהו יום חג, חול מועד או יום עסקים רגיל, וכדומה. מאפייני המימד משמשים את המשתמשים בחיתוך המידע על פי דרישותיהם. למשל, הצגת המכירות בחנויות החדשות הממוקמות בקניונים בלבד.

המבנה המתקבל פשוט מאוד להבנה. במרכזו נמצאות העובדות הגולמיות (מה היו המכירות בש"ח, מה כמות שנמכרה) ומסביב לעובדות מוצגים המימדיים העסקיים (מוצר, חנות, מבצע, צינור הפצה, זמן). ברגע שמציגים מבנה זה למשתמש, ונותנים לו כלי בסיסי המאפשר ביצוע שאילתות מול מבנה זה, מהר מאוד נמצא שהמשתמשים מתחילים לשאול שאלות בעלות עניין רב, מבחינת החברה. לדוגמה, ניתן לשאול שאילתות כגון:

- מה היה סך כל המכירות של טלוויזיות סוני דגם 123, בסניף מפרץ חיפה, במהלך כל חודש פברואר.
- מה היה סך כל המכירות של טלוויזיות סוני דגם 123, כתוצאה ממבצע ההנחות של 20%.
- מה היה סך כל המכירות של חנות מפרץ חיפה בחודשים ינואר ובפברואר.

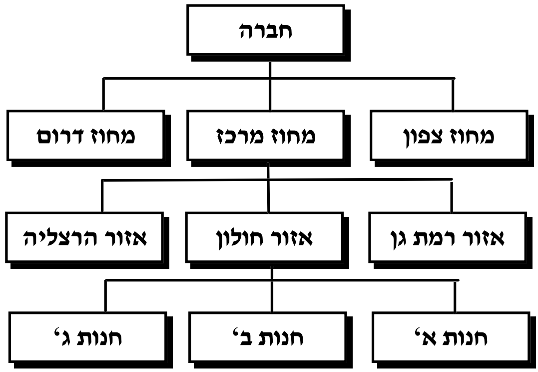
מובן שאלה רק מספר דוגמאות לשאילתות, מתוך מיגוון גדול מאוד של שאילתות שיכולות לעניין את מי שעוסק בניתוח מכירות החברה. בדרך כלל מעוניינים מקבלי החלטות בניתוח מגמות, וביצוע השוואות בין מימדים שונים. דוגמאות לשאילתות מורכבות יותר המבוססות על אותו מבנה פשוט:

- מהו אחוז השינוי במכירות של טלוויזיות סוני דגם 123, בפברואר השנה לעומת פברואר שנה שעברה.
- מהם חמשת המוצרים הנמכרים ביותר בחנות מפרץ חיפה, במהלך המחצית הראשונה של השנה.
- באילו מוצרים היתה ירידה של מעל 10% בהיקף המכירות במהלך כל שנת העסקים.

ניתן לחשב ולקבל את התוצאות של כל שאילתות אלו, מכיון שהנתונים הגולמיים הדרושים לשם כך נמצאים במודל הרב-מימדי הפשוט שבנינו.

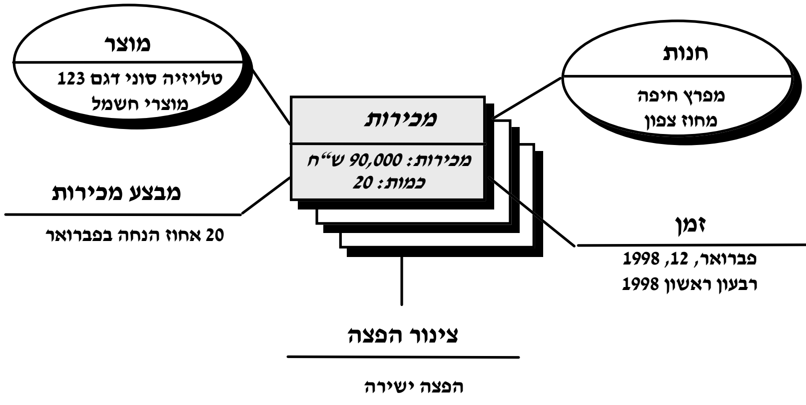
היררכיות בתוך מימדים

בדרך כלל, בתוך כל מימד קיימת היררכיה בעלת משמעות מבחינה עסקית. לדוגמה, נתאר ארגון בו יש חלוקה גיאוגרפית של הארץ לשלושה מחוזות - צפון, מרכז ודרום. כל מחוז מחולק למספר אזורים ולכל אזור יש מספר חנויות. זוהי היררכיה טבעית בתוך המימד הגיאוגרפי של החברה.



תרשים 4.6: היררכיה בתוך המימד הגיאוגרפי.

גם בתוך מימד המוצרים קיימת היררכיה - מוצר, קטגוריית מוצרים, קו מוצרים. דוגמה אחרת יכולה להיות ההיררכיה הטבעית הקיימת בתוך מימד הזמן - ימים, שבועות, חודשים, רבעונים, שנים. אם נוסיף עוד מספר נתונים למימדים נוכל לקבל מידע מעניין נוסף. למשל, אם נוסיף למימד הגיאוגרפי את ההיררכיה שלו, למימד הזמן את ההיררכיה של רבעון ולמימד המוצר עוד רמה היררכית כגון קטגוריית מוצרים, נוכל להתחיל לבצע שאילתות מורכבות יותר.



תרשים 4.7: הוספת היררכיה למימדים העסקיים.

להלן מספר דוגמאות לשאילתות שנוכל לבצע, כתוצאה מהוספת מאפייני ההיררכיות למימדים:

- ❖ מה היה סך כל המכירות במהלך הרבעון הראשון של שנת 1998, לכל המוצרים בקטגוריה "מוצרי חשמל".
- ❖ מהי קטגוריית המוצר שנמכרה טוב ביותר במחוז הצפון.
- ❖ מהו השינוי במכירות בין הקטגוריות "מוצרי חשמל" ו"כלי בית", בין רבעון ראשון של שנת 1997 לרבעון הראשון של שנת 1998.

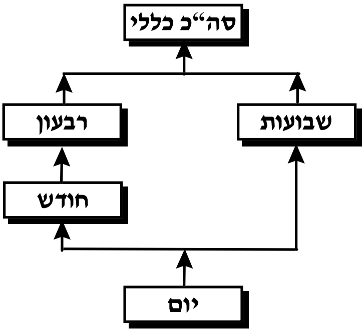
❖ מהם שלושת המרחבים שמכרו הכי הרבה בשני הרבעונים הראשונים של שנת 1998.

❖ מהו סך כל המכירות לפי קטגוריית מוצר, לפי מחוז ולפי רבעון.

בנוסף לשאילתות אלה ניתן גם לאפשר ניתוח What If. לדוגמה:

❖ בהנחה שברבעון ראשון של שנת 1998 היו המכירות גבוהות ב- 10% לעומת מה שהיה בפועל, הצג את הרבעון הטוב ביותר במהלך כל שנת 1998.

עד כאן, הצגנו מצב של היררכיה אחת בתוך המימד. במציאות, ניתן למצוא מצבים בהם קיימות מספר היררכיות שונות בתוך אותו מימד. מכיון שהיררכיה בתוך המימד מייצגת צורת סיכום מסוימת של הנתונים, העובדה שיתכנו מספר היררכיות מקבילות בתוך המימד אינה אמורה להפתיע אותנו, מכיון שאנשים שונים יכולים לבקש היררכיות סיכום שונות. נסתכל רגע במימד הזמן - משתמש אחד יכול לבקש לראות את המכירות לפי שבועות, ואילו אחר מבקש לראות את המכירות לפי חודשים ורבעונים. לא ניתן להגדיר היררכיה אחת, מכיון ששבועות אינם מסתכמים לחודשים ורבעונים - שבוע מסוים יכול להתפרס על פני שני חודשים, או שני רבעונים שונים. הדרך הנכונה לטפל בסוגיה זו היא להגדיר שתי היררכיות נפרדות.



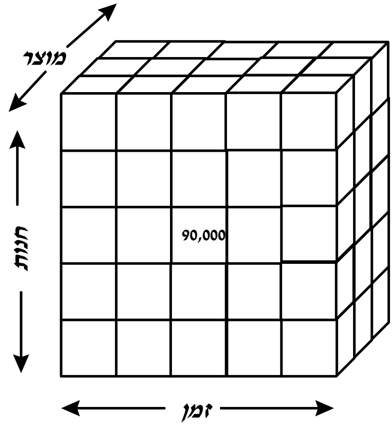
תרשים 4.8: ריבוי היררכיות בתוך מימד.

מתוך הדוגמאות הקודמות ניתן להבין את עוצמתו של המודל הפשוט שבנינו. באמצעות כלי התוכנה המכיר את המבנה, ומכיל אופרטורים מיוחדים לסוג זה של ניתוחים, ניתן לבצע חיתוכים וסיכומים מגוונים מאוד, תוך הפעלת פילטרים שונים על המימדים או על העובדות, וכל זאת ללא צורך לבקש ממנהל בסיס הנתונים לבנות משפטי SQL מורכבים.

הקוביה הרב-מימדית

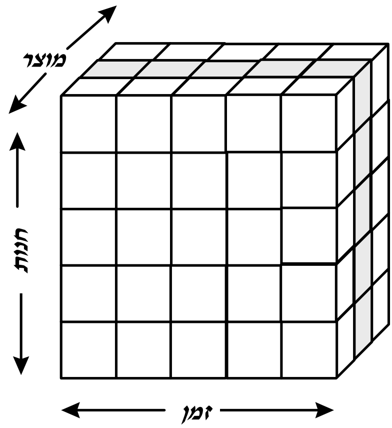
צורה נוחה לדמיין את המודל הרב-מימדי היא כקוביה רב-מימדית, כאשר כל פיאה שלה מייצגת מימד עסקי מסוים. בנקודת המפגש בין כל המימדים נמצאות העובדות המאוחסנות בתאים. המשתמש יכול לסובב את הקוביה הרב-מימדית לכל כיוון שהוא רוצה, ולנוע בין התאים השונים, או בתוך הקוביה כלפי מטה, לקבלת פירוט נוסף. לחילופין, הוא יכול לנוע כלפי מעלה, אם הוא רוצה לקבל מבט ברמת סיכום גבוהה יותר.

התרשים הבא מציג קוביה תלת מימדית ובה שלושת המימדים הם מימד הזמן, מימד החנות ומימד המוצר. בכל נקודת מפגש של שלושת המימדים מנוהל סך המכירות למוצר, לחנות, לתאריך. לדוגמה, בתא מסוים נמצא את המספר 90,000 המייצג את העובדה שסך המכירות לטלוויזיות סוני דגם 123, בחנות במפרץ חיפה, בתאריך 12/2/98, היו 90,000 ש״ח.



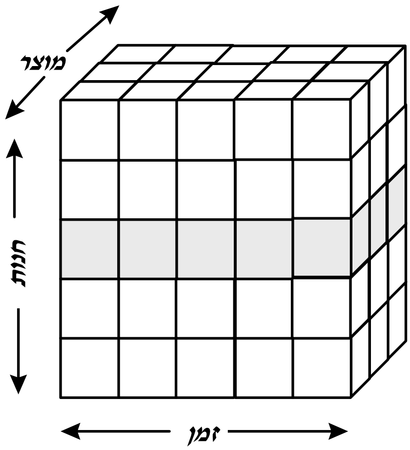
תרשים 4.9: קוביה תלת-מימדית.

קל לדמיין קוביה בעלת שלושה מימדים. קשה יותר לדמיין ארבעה, חמישה או יותר מימדים. למרות הקושי לדמיין קוביות רב-מימדיות אלו, ניהולן במחשב הוא בר-ביצוע. ואכן, קיימות מערכות תוכנה המסוגלות לתמוך במספר רב של מימדים בקוביה אחת. הקוביות הדמיוניות המתקבלות נקראות לעיתים גם Hypercube או n-Dimensional Cube. למרות שלקוביה יכולים להיות מספר רב של מימדים, המשתמשים מתמקדים תמיד במספר מצומצם של מימדים. למשל, מנהל מוצר מסוים מתעניין רק ב**פרוסה** (Slice) המכילה את מכירות המוצר שלו לאורך הזמן בחנויות השונות.



תרשים 4.10: נקודת מבט של מנהל המוצר.

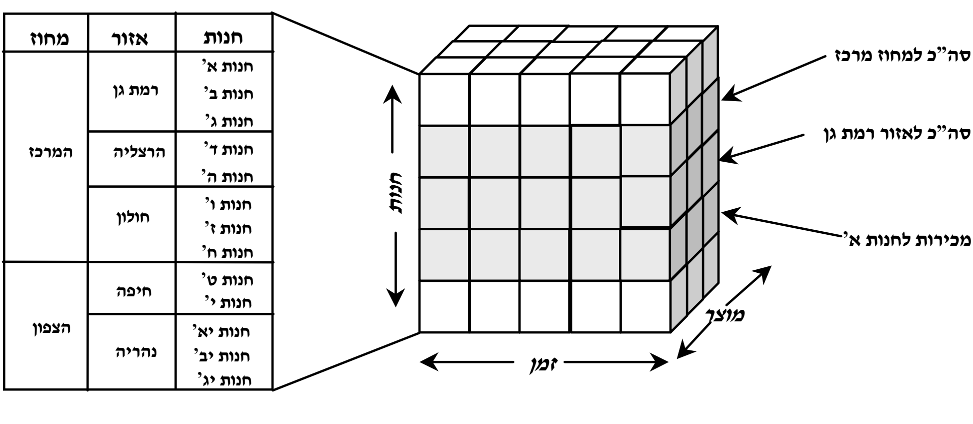
לעומת מנהל המוצר, למנהל החנות נקודת מבט שונה. הוא מעוניין בפרוסה המכילה את המכירות של כל המוצרים בחנות שלו לאורך הזמן.



תרשים 4.11: נקודת המבט של מנהל החנות.

כפי שניתן להבין משתי דוגמאות אלה, ניתן ליצור מספר בלתי מוגבל של פרוסות מידע, המציגות את המידע בקוביה הרב-מימדית בהתאם לנקודת המבט המבוקשת על ידי משתמשים שונים.

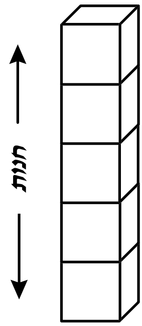
כפי שראינו, מימד יכול להכיל גם היררכיה פנימית. נתבונן לרגע במימד החנויות המכיל את ההיררכיה מרחב, אזור, חנות. בנוסף לתאים המכילים את המידע ברמת הפירוט הנמוכה ביותר, רמת החנות, ניתן לנהל שורות נוספות המכילות את הסיכומים עבור הרמות ההיררכיות הגבוהות יותר. בתרשים הבא מוצגת הקוביה עם שלוש רמות היררכיה שונות, שורת המכירות של חנות א', שורת הסיכומים של כל החנויות השייכות לאזור רמת גן ושורת הסיכומים של כל המכירות בכל האזורים השייכים למרחב המרכז.



תרשים 4.12: קוביה רב-מימדית עם היררכיות.

על פי בקשת המשתמש ניתן להציג או להסתיר את השורות השונות ולהציג רק את השורות המבוקשות - למשל רק את רמת החנויות באזור מסוים, רק את רמת האזורים, רק את רמת המרחבים, או כל שילוב ביניהם. לדוגמה, ניתן לבקש לראות את המכירות לפי מוצר, ובתוך היררכיית החנויות להציג את הסיכומים רק למרחבים או מרחבים ואזורים או מרחבים, אזורים וחנויות.

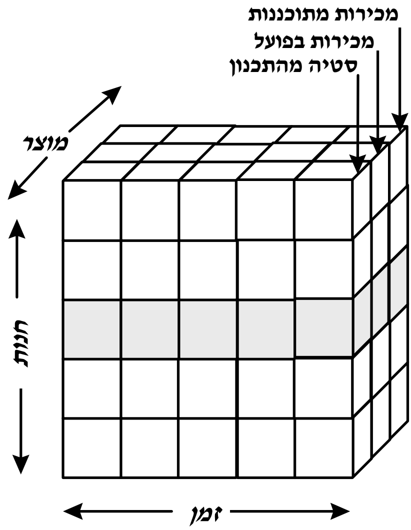
בנוסף ל**מבט המפורט** (Detailed View) ניתן להציג **נקודות מבט סיכומיות** (Aggregate View), בהן מסוכם מימד אחד או יותר. למשל, אם נרצה לקבל את סך כל המכירות לפי חנויות, נצטרך לסכם את מימד הזמן ואת מימד המוצר, עבור כל אחת מהחנויות. במקרה זה נקבל טבלה פשוטה בה מוצגות החנויות וליד כל חנות מה סך כל המכירות שלה, לכל המוצרים ולכל תקופות הזמן.



תרשים 4.13: מבט המציג סיכום מכירות לפי חנויות.

ניהול שורות הסיכום יכול להתבצע בצורות שונות - חישוב הסיכומים מראש בעת בניית הקוביה וניהולם בקוביה, או חישוב הסיכומים רק בעת בקשה להצגתם. לכל שיטה יתרונות וחסרונות, בכל הקשור לנפח הנתונים שיש לנהל ולזמן התגובה להצגת הנתונים.

בנוסף לעמודות והשורות המנוהלות בקוביה הרב-מימדית, ניתן ליצור שורות או עמודות חדשות, הנובעות ממניפולציה מתמטית כלשהי המבוצעת על הנתונים. עמודות או שורות אלו נקראות **עמודות / שורות מחושבות** (Derrived Data). הן נקראות כך מכיון שאינן מנוהלות באופן ישיר בקוביה, אלא מחושבות בעת הצורך. למשל, אם יש לנו מכירות בפועל ומכירות מתוכננות, ניתן בקלות להוסיף מערך נוסף, המכיל את ההפרש בין התכנון לבין הביצוע, ולכן הוא מציג את הסטיות מהתכנון. רוב כלי הניתוח הרב-מימדים כוללים אופרטורים רבי עוצמה לביצוע חישובים על וקטורים (Vector Arithmetic).



תרשים 4.14: מערכים מחושבים.

שיטת העבודה עם אופרטורים אלה מזכירה את העבודה עם גיליון אלקטרוני, בו יש תאים עם נתונים ותאים עם נוסחאות. בזמן הצגת הגיליון האלקטרוני מתבצע חישוב מיידי של כל הנוסחאות, כך שהמשתמש אינו מבחין בהבדל בין נתונים רגילים לבין נתונים תוצאתיים. ההבדל העקרוני בין הגיליון האלקטרוני ובין מערכות רב-מימדיות הוא שהגיליון האלקטרוני יודע לטפל בטבלאות דו-מימדיות בלבד, בעוד שהמערכות הרב-מימדיות מרחיבות יכולות אלו לטיפול במספר גדול מאוד של מימדים. כדי לשמור על זמני תגובה מהירים בזמן הניווט במבנה הרב-מימדי, נדרשת רמת תחכום גבוהה בצורת ניהול הנתונים, בניהול אינדקסים מהירים ובצורת ביצוע החישובים. חלק מהחישובים מבוצעים מראש, בעת טעינת הקוביה בנתונים, וחלק מבוצעים תוך כדי העבודה עם הקוביה (On the fly).

גמישות נרכשת זו של התפיסה הרב-מימדית, והתאמתה הרבה לסוג השאלות שמנתחי מידע ומקבלי החלטות שואלים בדרך כלל, הביאה לפופולריות הרבה של מערכות התומכות במבט רב-מימדי בנתונים. אין כל ספק שהתפיסה הרב-מימדית מאפשרת ייצוג טוב ונוח יותר של נקודות המבט העסקיות, מאשר התפיסה הטבלאית, המציגה הכל כאוסף טבלאות דו-מימדיות ומאפשרת את ניתוח הנתונים באמצעות פקודות SQL. נקודות המבט של המימדים מאפשרות למשתמשים לנווט את דרכם דרך כל המימדים, ודרך כל הרמות. המגמה בניווט מסוג זה היא לגלות מגמות החבויות בנתונים, ולהפוך אותן להחלטות עסקיות רבות חשיבות.

הצגה מימדית של מידע (Dimensional View)

השיטה הרגילה להצגת הנתונים היא במבנה טבלה דו-מימדית המורכבת מעמודות ושורות. זו השיטה בה מציגים בדרך כלל בסיסי הנתונים הטבלאיים את הנתונים. הטבלה הבאה מציגה מכירות מוצרים לפי יום בשבוע.

מוצרים	יום בשבוע	סה"כ מכירות
מוצר א'	יום א'	2,945
מוצר א'	יום ב'	3,035
מוצר ב'	יום א'	1,910
מוצר ב'	יום ב'	2,484

תרשים 4.15: מבנה טבלאי רגיל.

למרות שטכניקת הצגת נתונים זו פשוטה להבנה, ניתן להציג את המידע המופיע בטבלה שבתרשים הקודם בצורה נוחה יותר, והיא השיטה המימדית. בשיטה זו מופיעים המימדים בשני צירי הטבלה, בעוד הנתונים העובדתיים מופיעים במבנה. שיטה זו מזכירה את מבנה הגיליון האלקטרוני. העמודות במבנה הטבלאי הופכות כאן למימדים, ואילו השורות במבנה הטבלאי הופכות לתאים. לעובדות המופיעות בתאים מקובל לקרוא גם בשם **משתנים** (Variables). בדוגמה הבאה מוצגים מימדי הזמן והמוצר.

	מוצר א'	מוצר ב'
יום א'	2,945	1,910
יום ב'	3,035	2,484

תרשים 4.16: טבלה רב-מימדית המציגה שני מימדים - זמן ומוצר.

כפי שניתן לראות, זו שיטה קומפקטית ונוחה יותר להבנה.

הקוביה הרב-מימדית היא מודל קונצפטואלי נוח, אולם קיים קושי להציג אותה בנוחיות. הדרך הנוחה ביותר להצגת נתונים רב-מימדיים היא על ידי הצגתם במבנה טבלה דו-מימדית, כאשר כל פעם מחליפים את המימדים המוצגים. פעם אחת ניתן להציג את הנתונים לפי המוצרים והימים, פעם אחרת את המוצרים לפי חנויות, פעם אחרת את החנויות לפי ימים, וכן הלאה. מכאן, שאחת הפעולות הבסיסיות ביותר בניתוח הרב-מימדי היא בחירת המימדים המוצגים. בחירה זו מתבצעת, בדרך כלל, על ידי הצגת תפריט המכיל את כל המימדים המוצגים. כעת יש להצביע על צמד המימדים שיש להציג או לגרור את המימדים המבוקשים לשטח התצוגה.

לפני

	מוצר א'	מוצר ב'
יום א'	2,945	1,910
יום ב'	3,035	2,484

החלפת מימדים

אחרי / לפני

	חנות א'	חנות ב'
מוצר א'	1,423	1,450
מוצר ב'	2,520	2,300

החלפת מימדים

אחרי

	מוצר א'	מוצר ב'
חנות א'	2,135	2,310
חנות ב'	3,210	1,756

תרשים 4.17: מבט רב-מימדי כאוסף של מבטים דו-מימדיים.

הצגת הנתונים המימדית אינה מוגבלת להצגת שני מימדים בלבד. ניתן להציג בו-זמנית מספר רב יותר של מימדים בצורה **מקוננת** (Nested Dimensions). הסדר בו מופיעים המימדים המקוננים נקבע על ידי המשתמש. בתרשים הבא בחר המשתמש לראות שני מימדים אנכיים מקוננים - מוצרים ובתוכם את מבצעי ההנחות ושני מימדים אופקיים מקוננים - ימים ובתוכם את החנויות.

		מוצר א'		מוצר ב'	
		מבצע הנחות א'	מבצע הנחות ב'	מבצע הנחות א'	מבצע הנחות ב'
יום א'	חנות א'	520	630	210	125
	חנות ב'	645	220	420	600
	חנות ג'	830	110	320	225
יום ב'	חנות א'	340	445	645	450
	חנות ב'	910	124	710	340
	חנות ג'	210	900	220	225

תרשים 4.18: טבלה עם ארבעה מימדים מקוננים.

מובן שהמשתמש יכול לבחור לשנות את סדר הצגת המימדים המקוננים, למשל לבחור במימד מבצע ההנחות ובתוכו את המוצרים. התרשים הבא מציג את הטבלה לאחר השינוי בסדר הצגת המימדים המקוננים.

		מבצע הנחות א'		מבצע הנחות ב'	
		מוצר א'	מוצר ב'	מוצר א'	מוצר ב'
יום א'	חנות א'	520	210	630	125
	חנות ב'	645	420	220	600
	חנות ג'	830	320	110	225
יום ב'	חנות א'	340	645	445	450
	חנות ב'	910	710	124	340
	חנות ג'	210	220	900	225

תרשים 4.19: טבלה עם ארבעה מימדים מקוננים לאחר שינוי סדר הקינון.

ניתן להציג מספר רב יותר של מימדים מקוננים בטבלה אחת. התרשים הבא מציג כותרות של שלושה מימדים אנכיים מקוננים - מבצעי הנחות, מוצרים, חנויות.

ברור שככל שנציג יותר מימדים, הופכת הטבלה למורכבת יותר, וקשה יותר להציג אותה במסך.

מבצע הנחות א'						מבצע הנחות ב'					
מוצר א'			מוצר ב'			מוצר א'			מוצר ב'		
חנות א'	חנות ב'	חנות ג'	חנות א'	חנות ב'	חנות ג'	חנות א'	חנות ב'	חנות ג'	חנות א'	חנות ב'	חנות ג'

תרשים 4.20: שלושה מימדים אנכיים מקוננים

פעולות יסוד בניתוח מידע רב-מימדי

אחד היתרונות הגדולים של המוצרים התומכים בניתוח מידע רב-מימדי הוא העושר באופרטורים המאפשרים למשתמש לבצע מיגוון רחב של פעולות בקוביה, בצורה ידידותית וקלה מאוד. אופרטורים אלה שונים באופן מהותי מהאופרטורים הרגילים של שפת SQL, באמצעותה מקובל לפעול מול בסיסי הנתונים הטבלאיים. באמצעות אופרטורים אלה יכול המשתמש לפעול בקוביה הרב-מימדית ולבצע, בין היתר, פעולות כגון בחירת המימדים להצגה, סיבוב הקוביה על ידי החלפת המימד האנכי והאופקי, פריסה וחיתוך הקוביה, קידוח מטה לקבלת רמת מידע מפורטת יותר ועוד. מכיון שרוב המוצרים התומכים בניתוח הרב-מימדי פועלים בסביבה חלונאית, הם משתמשים בכל מיגוון האפשרויות שסביבה זו מעמידה לרשות המשתמש - תפריטים נפרשים, שיטות הצבעה וגרירה (Drag and Drop), אובייקטים ויזואליים שונים וכדומה. רוב המוצרים עושים גם שימוש נרחב בגרפיקה עסקית להצגת המידע באופן גרפי, כאשר המשתמש שולט בגרף הרצוי (גרף מוטות, גרף לינארי, גרף עוגה וכדומה). כעת, נסקור בקצרה את האופרטורים הנפוצים ביותר בניתוח הרב-מימדי, אופרטורים הנתמכים על ידי רוב המוצרים המסחריים.

- ❖ **בחירת מימדים** (Dimension Selection): הפעולה הבסיסית ביותר היא כמובן בחירת המימדים להצגה. בחירה זו יכולה להתבצע בצורות שונות כאשר הנפוצה ביותר היא על ידי בחירה מתפריט מיוחד.
- ❖ **החלפת מימדים** (Dimension Switching): פעולה המאפשרת למשתמש להחליף את המימדים המוצגים. הפעולה מאפשרת בחירת מימדים חדשים, או הוספת מימדים למימדים המוצגים, לקבלת הצגה מקוננת. רוב הכלים מאפשרים ביצוע פעולת שינוי סדר הצגת המימדים על ידי פעולה פשוטה של גרירה (Drag and Drop).
- ❖ **סיבוב מימדים** (Rotate): פעולת הסיבוב מאפשרת החלפה בין המימדים האנכיים והאופקיים. לדוגמה, אם המימד האנכי הוא **מוצרים** והמימד האופקי הוא **זמן**, נוכל לבצע פעולת סיבוב, כך שהמימד האנכי יהיה המימד **זמן** והמימד האופקי יהיה המימד **מוצרים**.



לפני

	מוצר א'	מוצר ב'
יום א'	2,945	1,910
יום ב'	3,035	2,484

אחרי

	יום א'	יום ב'
מוצר א'	2,945	3,035
מוצר ב'	1,910	2,484

תרשים 4.21: סיבוב המימדים.

❖ **סיבוב סביב ציר** (Pivoting): פעולה המאפשרת לבחור נתון מסוים, להגדיר אותו כציר (Pivot) ולהציג אותו במיגוון גדול של נקודות מבט שונות. למשל, ניתן לבחור במכירות של חנות מסוימת ולהתבונן בנתון מנקודת המבט של חנויות נוספות, מנקודת מבט של מוצרים, מנקודת מבט של זמן וכדומה. פעולה זו דומה לפעולת סיבוב המימדים, אבל מרחיבה אותה במובן שניתן לסובב את הנתון סביב כל המימדים האפשריים, לא רק על ידי החלפת המימד האנכי במימד האופקי.

❖ **פריסה וחיתוך** (Slicing and Dicing): פעולה זו מאפשרת להתבונן במבנה הנתונים הרב-מימדי מנקודות מבט שונות. כלומר, פעולה המאפשרת את החלפת המימדים המוצגים, על פי הצורך. ניתן להתייחס אל כל פרוסה כאל תת-קבוצה של הקוביה, כך שערכים של מימד (או מימדים) מסוימים נשמרים קבועים. לדוגמה, ניתן להגדיר פרוסה של יום מסוים, ואז להציג את המכירות באותו יום של כל המוצרים לפי חנויות, או את המכירות באותו יום לפי חנויות ומבצעי הנחות.

	מוצר א'	מוצר ב'
חנות א'	1,850	2,100
חנות ב'	4,130	2,294

תרשים 4.22: החלפת מימדים.

❖ **חישוב נתונים** (Calculated Data): אחת הפעולות הבסיסיות ביותר המאפשרת לנו להציג עמודות או שורות הנובעות מחישוב כלשהו ולאו דווקא נתונים המנוהלים בבסיס הנתונים. הפעולה הנפוצה ביותר היא ביצוע סיכומים של שורות או עמודות. בדוגמה הבאה הוספנו סיכום עמודה המציגה את סך כל המכירות לחנות לפי מוצר, לפי חנות והסך הכל הכללי.

	מוצר א'	מוצר ב'	סה"כ לחנות
חנות א'	1,850	2,100	3,950
חנות ב'	4,130	2,294	6,424
סה"כ למוצר	5,980	4,394	10,374

תרשים 4.23: טבלה דו-מימדית עם סיכומים.

אמנם הפעולה הנפוצה ביותר היא סיכומים, אבל באותה מידה ניתן להפעיל נוסחה מתמטית כלשהי לקבלת העמודה המחושבת, בדומה למקובל בגיליון אלקטרוני. כשמבצעים פעולות סיכום בהיררכיה של מימד, כמו למשל סיכום המכירות של חנות לאזור, מקובל לקרוא לפעולה זו גם בשם Consolidation או Roll Up. כדי לשמור על זמני תגובה מהירים, מקובל לבצע חלק מהחישובים מראש, ולשמור אותם בקוביה.

❖ **קידוח מטה (Drill Down) בהיררכיית המימד:** פעולה המאפשרת לרדת כלפי מטה בהיררכיית מימד לקבלת רמת פירוט גבוהה יותר. זו אחת הפעולות הבסיסיות ביותר בניתוח רב-מימדי המתחיל בדרך כלל במבט-על כלשהו, מבט המאפשר לראות מגמות או חריגים, ומשם מתחיל תהליך הקידוח כדי לקבל פירוט נוסף על פי הצורך.

נתחיל בהדגמת תהליך הקידוח על ידי פירוט המבנה ההיררכי במימד החנויות. נזכיר עוד שמימד זה מכיל את הרמות ההיררכיות הבאות: סה"כ כללי, סה"כ למרחב, סה"כ לאזור, סה"כ לחנות. הטבלה הבאה מציגה את סך כל המכירות למוצר א' ו-ב'.

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100

תרשים 4.24: טבלת סה"כ מכירות לפי מוצרים.

לאחר הצגת טבלה סיכומית זו, מבקש המשתמש לראות את פירוט המכירות לפי שלושת המחוזות. בדרך כלל, הקידוח מטה מתבצע על ידי לחיצה כפולה בעכבר, כאשר הסמן מצביע על השורה המבוקשת. באמצעות הפעלת אופרטור הקידוח, הוא יקבל את הטבלה הבאה:

פירוט לפי מחוזות

לפני

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100

אחרי

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100
סה"כ למחוז הצפון	17,800	10,920
סה"כ למחוז המרכז	22,450	18,250
סה"כ למחוז הדרום	14,960	15,930

תרשים 4.25: קידוח מטה לקבלת פירוט לפי מחוז.

במצב זה יכול המשתמש לבקש רמת פירוט נוספת, לפי האזורים בכל המחוזות או רק למחוז מסוים. נניח, שהמשתמש ביקש לראות פירוט נוסף - לפי אזורים במחוז המרכז בלבד. הוא יקבל את הטבלה המופיעה בתרשים 4.26.

פירוט לפי אזורים

לפני

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100
סה"כ למחוז הצפון	17,800	10,920
סה"כ למחוז המרכז	22,450	18,250
סה"כ למחוז הדרום	14,960	15,930

אחרי

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100
סה"כ למחוז הצפון	17,800	10,920
סה"כ למחוז המרכז	22,450	18,250
סה"כ לאזור רמת גן	7,450	10,970
סה"כ לאזור חולון	9,450	6,730
סה"כ לאזור הרצליה	2,000	550
סה"כ למחוז הדרום	14,960	15,930

תרשים 4.26: קידוח מטה לקבלת פירוט לפי אזורים בתוך מחוז.

מהתמונה המתקבלת ניתן לראות שהמכירות באזור הרצליה נמוכות יחסית לאזורים האחרים, ולכן יכול המשתמש לבקש לראות פירוט נוסף, לפי החנויות באזור הרצליה. הוא יקבל את הטבלה הבאה.

פירוט לפי חנויות

לפני

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100
סה"כ למחוז הצפון	17,800	10,920
סה"כ למחוז המרכז	22,450	18,250
סה"כ לאזור רמת גן	7,450	10,970
סה"כ לאזור חולון	9,450	6,730
סה"כ לאזור הרצליה	2,000	550
סה"כ למחוז הדרום	14,960	15,930

אחרי

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100
סה"כ למחוז הצפון	17,800	10,920
סה"כ למחוז המרכז	22,450	18,250
סה"כ לאזור רמת גן	7,450	10,970
סה"כ לאזור חולון	9,450	6,730
סה"כ לאזור הרצליה	2,000	550
סה"כ לחנות א'	650	320
סה"כ לחנות ב'	920	120
סה"כ לחנות ג'	430	110
סה"כ למחוז הדרום	14,960	15,930

תרשים 4.27: קידוח מטה לקבלת פירוט לפי חנויות בתוך אזור.

כפי שניתן לראות, כמות הנתונים שיש להציג הולכת וגדלה. לכן, ניתן להגביל את הנתונים המוצגים רק לשורות או העמודות המבוקשות. בתרשים הבא מוצגת הטבלה לאחר שהמשתמש ביקש לראות רק את השורות המתייחסות לאזור הרצליה.

	מוצר א'	מוצר ב'
סה"כ לאזור הרצליה	2,000	550
סה"כ לחנות א'	650	320
סה"כ לחנות ב'	920	120
סה"כ לחנות ג'	430	110

תרשים 4.28: הצגת הטבלה המפורטת לאחר צמצום השורות.

- **קידוח מטה (Drill Down) על ידי הוספת מימד נוסף:** ניתן לבקש פירוט נוסף לא רק על ידי ירידה בהיררכיית המימד, אלא גם על ידי הוספת מימד מקונן נוסף. הפעם, נתחיל את פעולת הקידוח לאחר שקיבלנו את הטבלה המציגה את המכירות, לפי מוצרים וזמן.

	מוצר א'	מוצר ב'
יום א'	2,945	2,730
יום ב'	3,035	2,484

תרשים 4.29: טבלת מכירות מוצרים לפי ימים.

נניח שבמצב זה, זיהה המשתמש בעיה מסוימת באחד הימים. לכן, ניתן לבקש להוסיף את מימד החנות למימד הזמן, כדי לאתר באיזו חנות נוצרה הבעיה. נקבל את הטבלה המוצגת בתרשים 4.30.

פירוט לפי חנויות

לפני

	מוצר א'	מוצר ב'
יום א'	2,945	2,730
יום ב'	3,035	2,484

אחרי

		מוצר א'	מוצר ב'
יום א'	חנות א'	730	755
	חנות ב'	1,065	820
	חנות ג'	1,150	1,155
יום ב'	חנות א'	985	895
	חנות ב'	1,620	464
	חנות ג'	430	1,125

תרשים 4.30: קידוח מטה על ידי הוספת מימד אופקי נוסף.

קבלת הפרטים יכולה להתבצע עד לרמה הנמוכה ביותר הנדרשת. נוכל לבקש רמת פירוט נוספת על ידי הוספת מימד נוסף למימדים האנכיים או האופקיים שכבר מוצגים. בדוגמה שלנו, נוכל לבקש להוסיף את מימד מבצע ההנחות למימד האנכי של מוצרים.

פירוט לפי מבצעים

לפני

		מוצר א'	מוצר ב'
יום א'	חנות א'	730	755
	חנות ב'	1,065	820
	חנות ג'	1,150	1,155
יום ב'	חנות א'	985	895
	חנות ב'	1,620	464
	חנות ג'	430	1,125

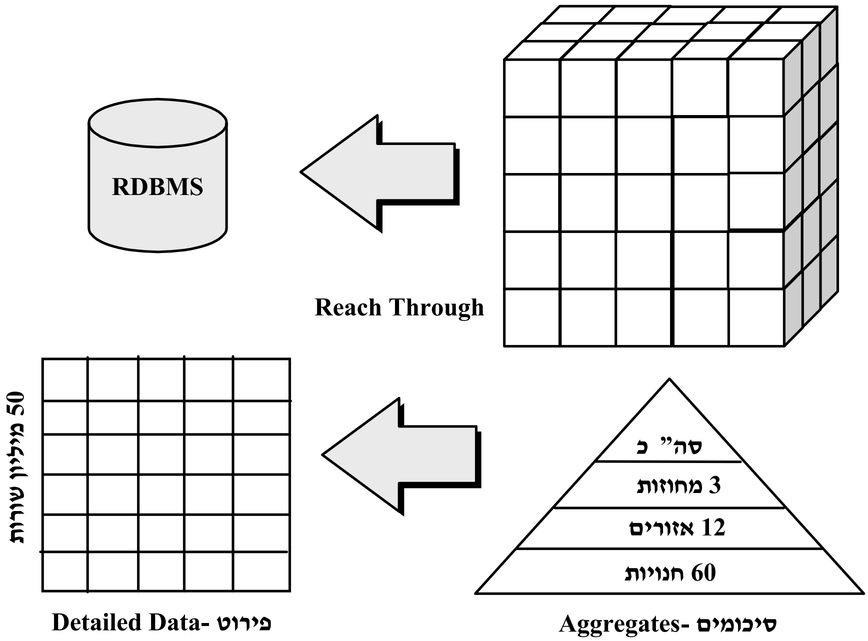
אחרי

		מוצר א'		מוצר ב'	
		מבצע הנחות א'	מבצע הנחות ב'	מבצע הנחות א'	מבצע הנחות ב'
יום א'	חנות א'	520	630	210	125
	חנות ב'	645	220	420	600
	חנות ג'	830	110	320	225
יום ב'	חנות א'	340	445	645	450
	חנות ב'	910	124	710	340
	חנות ג'	210	900	220	225

תרשים 4.31: קידוח מטה על ידי הוספת מימד אנכי נוסף.

❖ **קבלת פירוט מעבר למנוהל בבסיס הנתונים הרב-מימדי** (Reach Through): לעיתים, מבקש מנתח המידע לקבל פירוט, מעבר לרמה הנמוכה ביותר הנשמרת במערכת הרב-מימדית. למשל, לאחר בדיקה של מכירות בחנות מסוימת בתאריך כלשהו, הוא מבקש לקבל מידע לגבי המכירות לפי כל שעה ביום, מידע שאינו נשמר במערכת הרב-מימדית. במקרה זה, יש לבצע פעולת מעבר מהמערכת הרב-מימדית אל בסיס נתונים טבלאי, על ידי שיגור פקודות SQL מתאימות. רוב המערכות המודרניות מחוללות את פקודות SQL באופן אוטומטי, ושקוף למשתמש.

תרשים 4.32 מציג את המעבר מהקוביה הרב-מימדית אל בסיס הנתונים הטבלאי, המנהל את כל הנתונים ברמה הפרטנית. כפי שניתן לראות, הקוביה הרב-מימדית מנהלת מספר מצומצם יחסית של סיכומים, בעוד שבסיס הנתונים הטבלאי מנהל מיליוני שורות.



תרשים 4.32: מעבר מרמה סיכומית לרמת פירוט בבסיס נתונים טבלאי.

❖ **קידוח מעלה** (Drill Up): זו פעולה הפוכה לפעולת הקידוח. היא מאפשרת מעבר מרמה מפורטת כלשהי לרמת סיכום גבוהה יותר. לדוגמה, סיכום המכירות מרמת החנות הבודדת לרמת האזור, ומשם לרמת המחוז ועד לסיכום הכללי. בפעולה זו אנו מתחילים מרמה כלשהי בהיררכיית המימד, ועולים כלפי מעלה. לעיתים יש צורך בביצוע פעולות אריתמטיות מורכבות יותר מאשר רק חיבור אלגברי.



לפני:

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100
סה"כ למחוז הצפון	17,800	10,920
סה"כ למחוז המרכז	22,450	18,250
סה"כ למחוז הדרום	14,960	15,930

אחרי:

	מוצר א'	מוצר ב'
סה"כ מכירות	55,210	45,100

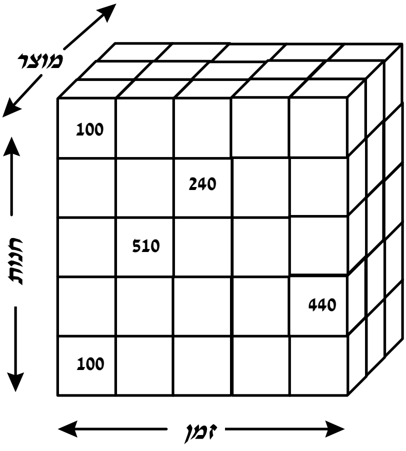
תרשים 4.33: מעבר מרמת פירוט לרמה סיכומית גבוהה יותר.

❖ **הגדרת המידע המוצג על ידי אופרטורים קבוצתיים** (Set Operators): פעולה זו מאפשרת למשתמש לבחור את המידע המוצג, תוך שימוש באופרטורים מתורת הקבוצות. אלה הם אופרטורים כגון And, Or, Not, Union, Intersect וכדומה. למשל, המשתמש יכול לבקש לראות את המכירות בחנויות חדשות, הממוקמות בקניונים בלבד באזור חיפה ורמת גן. ניתן גם להגדיר קבוצות שונות, ולהפעיל את האופרטורים על קבוצות אלה.

מגבלות גודל הקוביה הרב-מימדית

מקובל לחשוב שהקוביות הרב-מימדיות מוגבלות במספר המימדים שהן מסוגלות לטפל. למרות שקיימות מגבלות במספר מימדים, המגבלה החשובה יותר שיש לתת עליה את הדעת היא מספר התאים בקוביה. ככל שמספר המימדים גדל, גדל גם מספר התאים בקוביה אותם יש לנהל. אם בפנינו שני מימדים, מימד **מוצרים**, עם 500 מוצרים שונים, ומימד **זמן**, עם 52 שבועות - בפנינו קוביה ובה 26,000 תאים. אם נוסיף מימד נוסף, **חנויות**, ובו 300 חנויות שונות גדל מספר התאים וצומח עד 7,800,000 תאים. נוסיף מימד נוסף, מבצעי מכירות, שגודלו 50 והקוביה שלנו צמחה לגודל 39,000,000 תאים. כפי שניתן לראות, קוביה עם עשרה מימדים שונים יכולה להגיע בקלות למספר מיליארדים של תאים. רוב המערכות המסחריות, התומכות בקוביות רב-מימדיות, מגבילות את מספר התאים. לכן, על מעצב המערכת לשקול היטב מהו מספר המימדים, וכמה ערכים שונים יכול לקבל כל מימד, כדי לבחון האם יש גלישה ממגבלות גודל הקוביה.

סוגיה מעניינת נובעת מהעובדה שלא כל התאים האלה אכן מכילים נתונים. לדוגמה, לא כל מוצר נמכר בכל חנות בכל יום. חלק מהחנויות אפילו אינן מחזיקות מלאי של מוצרים מסוימים, ולכן הן כלל אינן מוכרות מוצרים אלה. בהנחה שזה אכן המצב, נמצא שאחוז גדול מכלל התאים בקוביה הרב-מימדית, יהיו למעשה ריקים. לקוביה בה מספר התאים הריקים גדול מקובל לקרוא **קוביה דלילה** (Sparse Cube).



תרשים 4.34: קוביה דלילה.

המערכות התומכות בניתוח מידע רב-מימדי יודעות לטפל במצבים אלה ואינן שומרות תאים ריקים, אלא רק תאים המכילים תוכן. שיטת ניהול זו של קוביות דלילות מאפשרת להתמודד עם מגבלות מספר המימדים והערכים.

סיכום

בפרק זה הראינו ששיטת ניתוח הנתונים הרב-מימדית היא שיטה נוחה מאוד לניתוח והצגת מידע למקבלי החלטות. בשל כך, היא מתאימה מאוד לסביבות של מחסני הנתונים. שיטה זו הפכה לפופולרית מאוד בשל הפשטות היחסית שלה, ויכולתה להציג את הנתונים בצורה נוחה, תוך שימוש במושגים עסקיים הברורים לכל משתמש.

תפיסת הניתוח הרב-מימדי מתאימה מאוד לאופי של ניתוח מידע לתמיכה בתהליכי קבלת החלטות, מכיון שהיא מאפשרת למשתמשים לנתח את המידע באופן אינטואיטיבי, מהיר וגמיש, תוך שימוש במושגים עסקיים מוכרים. תהליכי קבלת ההחלטות וניתוח המידע הם רב-מימדיים מטבעם, מכיון שנוח מאוד לארגן את הנתונים לפי מימדים עסקיים, לבצע את ניתוח הנתונים על סמך המימדים שלהם, להתבונן במידע מנקודות מבט שונות וברמות פירוט שונות. המשתמשים יכולים לנווט בבסיס הנתונים או על ידי פעולות קידוח לקבלת פירוט נוסף, או על ידי הגדרת נתון מסוים כ-Pivot, והתבוננות בנתון מנקודות מבט שונות. היכולות לבצע שאילתות על פי מימדים (לקביעת רמת הסיכום ושימוש בפילטרים המגדירים את המידע המתקבל), הן הפעולות היסודיות ביותר לניתוח מידע לקבלת החלטות. לדוגמה, אם במערכת מנוהלים שני מימדים, מוצר וזמן, יוכל המשתמש לראות את המידע בחתכים ובצירופים שונים של מימד המוצר, או מימד הזמן, או צירוף כלשהו של שניהם גם יחד. ניתן לראות מה היו המוצרים שנמכרו בשבוע מסוים, או לגבי מוצר מסוים - כמה יחידות ממנו נמכרו לאורך זמן. בדומה, יכול המשתמש להשוות את המכירות של שבוע אחרון עם השבוע המקביל בחודש קודם, רבעון קודם או שנה קודמת.

פרק 5: ארכיטקטורות של מוצרים לניתוח רב-מימדי - OLAP Architectures

מבוא

הניתוח הרב-מימדי, או OLAP כפי שהוא מכונה לעיתים קרובות, הפך לאחת משיטות ניתוח הנתונים הפופולריות ביותר, תוך מתן דגש לניתוח מידע ברמות סיכומיות. למשל, סך כל המכירות לפי סניף ואזור, סך כל דקות שיחה של לקוחות לפי יעדים, תקציב מול ביצוע וכדומה. ככל שמנתחי המידע למדו להעריך את עוצמת כלי הניתוח החדשים ולשאול שאלות מתוחכמות יותר, הם נתקלו במצב בו המידע הסיכומי לא ענה על הדרישות והם היו זקוקים למידע ברמה פרטנית יותר. למשל, כדי לקבל תמונה מדויקת יותר של התפלגות שיחות הלקוח לא מספיק המידע על סך כל דקות שיחה לפי יעדים, אלא צריך לרדת גם לרמת השעה ביממה בה בוצעה השיחה, ואולי אפילו עד רמת השיחה הבודדת. ניהול מידע ברמה פרטנית כל כך הוא מעבר ליכולות הרגילות של כלי הניתוח הרב-מימדיים. על רקע צורך זה בניתוח ותחקור מידע ברמה פרטנית מאוד, החלה להתפתח ארכיטקטורה נוספת של מוצרי OLAP. מוצרים שעבדו באופן ישיר עם בסיס הנתונים הטבלאי של מחסן הנתונים, וענו על הצורך בניתוח רב-מימדי ברמה פרטנית יותר ובנפחי נתונים גדולים מאוד.

אחת השאלות המעניינות היא האם כדי לספק מבט רב-מימדי על הנתונים (Multi Dimensional View) צריך לנהל את הנתונים בצורה פיסית מיוחדת, או שניתן לקבל מבט רב-מימדי גם תוך שימוש בבסיסי נתונים טבלאיים (שהם, כידוע, מבוססים על טבלאות דו-מימדיות פשוטות בלבד). במשך מספר שנים רב, היתה מקובלת הגישה שמוצר OLAP צריך להיות מבוסס על בסיס נתונים רב-מימדי, כלומר על מבנה נתונים ייעודי. עם ההתפתחות והתפוצה הרבה של בסיסי הנתונים הטבלאיים התברר שניתן לבנות מבט רב-מימדי גם על בסיסי נתונים טבלאיים, כלומר מתן מבט רב-מימדי לוגי על בסיס נתונים דו-מימדי פיסי. לכל אחת משתי ארכיטקטורות אלה יש יתרונות וחסרונות.

בשנים האחרונות החלו להתפתח מוצרים המשלבים את שתי התפיסות. מוצרים אלה מסוגלים לספק זמני תגובה מהירים, יחד עם יכולות ניתוח מורכבות, על ידי ניהול נתונים סיכומיים בבסיס נתונים ייעודי, במשולב עם היכולת לגשת למידע פרטני המנוהל בבסיס הנתונים הטבלאי. המציאות הוכיחה ששילוב בין שתי התפיסות הוא הפתרון הנכון.

מבין כל הטכנולוגיות המרכיבות את מרחב ניתוח המידע, זכתה טכנולוגיית הניתוח הרב-מימדי לפופולריות גדולה מאוד. עד כדי כך שאנשים נוטים לחשוב שניתוח רב-מימדי, ניתוח מידע ומחסן נתונים הם מושגים נרדפים, ולא כך הוא הדבר. לבלבול זה תורמת לא מעט העובדה שכלי הניתוח הרב-מימדי נקראים כלי OLAP, כלומר כלים לעיבוד אנליטי מקוון (שהיא הפעילות הכוללת לניתוח מידע בסביבת מחסן הנתונים), במקום להיקרא **כלי ניתוח רב-מימדי** (Multi Dimensional Analysis). בלבול זה החל בתקופה בה יצרני טכנולוגיה זו נאבקו ביצרני בסיסי הנתונים הטבלאיים, וביקשו להוכיח שניתוח רב-מימדי אינו בר מימוש על בסיס הנתונים הטבלאיים. הוא דורש בסיס נתונים במבנה מיוחד וייעודי - בסיס הנתונים הרב-מימדי.

אחת החברות המובילות באותה תקופה, Arbor Software (הנקראת כיום Hyperion Solutions), פנתה אל אבי המודל הטבלאי, E.F. Codd, וביקשה ממנו לבחון את הנושא. החברה פרסמה מאמר ובו התייחסותו של Codd לנושא. המאמר כלל 12 כללים הקובעים מהם המאפיינים של מוצר המבקש להיקרא **מוצר OLAP**. Codd הוא זה שנתן לטכנולוגיה זו את השם OLAP. השם בה להדגיש שזו טכנולוגיה חדשה, שמטרתה עיבוד אנליטי מקוון, לעומת **טכנולוגיית עיבוד התנועות המקוון** (OLTP), עבורה פותחו בסיסי הנתונים הטבלאיים. צמידות זו של הופעת המונח החדש OLAP יחד עם מאבק יצרני כלי הניתוח הרב-מימדיים מול יצרני בסיסי הנתונים הטבלאיים, גרם לכך שיצרני הכלים החלו לכנות את הטכנולוגיה **שלהם** OLAP. עם הזמן הוכיחו יצרני בסיסי הנתונים הטבלאיים ויצרני צד שלישי שניתן לספק מבט רב-מימדי **גם** על בסיסי נתונים טבלאיים. הם קראו לטכנולוגיה זו בשם ROLAP (Relational OLAP), ואילו לטכנולוגיה המבוססת על בסיסי נתונים רב-מימדיים קנייניים קראו בשם MOLAP (Multi Dimensional OLAP).

כדי להוסיף לבלבול הקיים בין המושגים, יש לא מעט אנשים שמבלבלים בין המונח OLAP לבין המונח **מחסן נתונים**. האמת היא שהניתוח הרב-מימדי הוא רק ענף מסוים של התהליכים האנליטיים לניתוח מידע. אלו, ברובם הגדול, מתבצעים בצורה מקוונת ומהווים חלק מתפיסת מחסן הנתונים, שהינה תפיסת קצה לקצה של כל תהליך אספקת הנתונים לתמיכה בקבלת החלטות. עקרונית ניתן לבנות מחסן נתונים שאינו מכיל כלי ניתוח מידע רב-מימדי, למרות שמצב זה יהיה די נדיר.

חלופות למימוש ארכיטקטורה למוצר OLAP

כתוצאה מהאפשרויות השונות למימוש המבט הרב-מימדי, התפתחו במשך השנים מספר ארכיטקטורות שונות, כאשר ההבדל העיקרי ביניהן מתבטא בשני פרמטרים עיקריים: היכן מנוהלים הנתונים, והיכן מתבצעים העיבודים.

נתבונן תחילה בנושא מיקום הנתונים. בעיקרון, קיימות שלוש חלופות שונות לאחסון הנתונים המשתתפים בניתוח הרב-מימדי:

- **בסיס נתונים רב-מימדי** (Multi Dimensional Database) - הנתונים מאוחסנים בבסיס נתונים רב-מימדי מיוחד המנוהל במחשב השרת. הנתונים הדרושים לניתוח נגזרים מבסיס נתונים טבלאי המנוהל במסגרת מחסן הנתונים, או נגזרים ישירות ממערכות תפעוליות. הנתונים הגזורים עוברים תהליך עיבוד, הכולל את סיכום הנתונים ברמות ובחתכים שונים וביצוע חישובים מיוחדים, במידה ונדרשים. בסיס הנתונים הרב-מימדי מבוסס על מבני נתונים של מערכים (Array) ואינדקסים מיוחדים, המנוהלים בדיסק המקומי ובחלק מהמקרים בזיכרון השרת, לשיפור הביצועים. בדרך כלל, בסיסי נתונים אלה מאפשרים למספר רב של משתמשים גישה **לקריאה בלבד** (Read Only), אך אינם מאפשרים עדכון בו-זמני על ידי מספר משתמשים.
- **בסיס נתונים טבלאי** (Relational Database) - הנתונים מאוחסנים בבסיס נתונים טבלאי רגיל במחשב שרת, אבל בעיצוב מיוחד (מבנה כוכב לא מנורמל), מכיון ששיטת העיצוב הרגילה של בסיסי נתונים טבלאיים אינה מתאימה לניתוח רב-מימדי. הנתונים והסיכומים מנוהלים במסגרת טבלאות רגילות של בסיס הנתונים. לנושא עיצוב בסיסי נתונים טבלאיים עבור ניתוחים רב-מימדיים מוקדש פרק שלם בהמשך.
- **מבנה נתונים מיוחד במחשב האישי** (Desktop Based File System) - הנתונים הדרושים לניתוח הרב-מימדי מנוהלים במחשב האישי המקומי (תחנת העבודה), במבנה קבצים מיוחד עבור הניתוחים הרב-מימדיים. בדרך כלל, מבנה נתונים זה איננו בסיס נתונים במובן האמיתי של המושג, אלא ממומש במערכת הקבצים של המחשב האישי. הנתונים הדרושים יכולים להיגזר ישירות מבסיס נתונים טבלאי, או מבסיס נתונים רב-מימדי, המנוהלים בשרת. גם במקרה זה, חלק מהנתונים יכולים להיות מנוהלים בדיסק המקומי וחלק מנוהל בזיכרון המחשב המקומי לשיפור הביצועים.

לכל אחת מהחלופות המוזכרות יש יתרונות וחסרונות, מבחינת היכולת לטפל בנפחי נתונים גדולים, מבחינת הביצועים, מבחינת כושר הגידול ועוד. בדרך כלל, יהיו הביצועים של בסיס הנתונים הרב-מימדי המנוהל בשרת או במחשב האישי טובים יותר מבסיסי נתונים טבלאיים, אבל בנפח נתונים מצומצם יותר.

הפרמטר השני במימוש ארכיטקטורה של מוצר OLAP מבוסס על ההחלטה היכן יבוצעו העיבודים הדרושים לצורך הניתוח הרב-מימדי. מעבר למורכבות הנובעת מעצם ניהול הנתונים בצורה יעילה ומתאימה לניתוח רב-מימדי, חשוב להדגיש שניתוח מידע רב-מימדי מבוסס גם על מספר רב של עיבודים, שחלקם יכולים להיות מורכבים מאוד. לדוגמה, השוואת מכירות לפי סניפים, בין שנים שונות, איתור חמשת הלקוחות הגדולים ביותר, חישוב אחוזי סטייה בין תכנון וביצוע תקציבי, וכדומה. חישובים אלה מבוצעים תוך כדי הניתוח, לא מראש. שלוש החלופות למקום ביצוע העיבודים הן:

- **במנוע רב-מימדי בשרת** (Multi Dimensional Server Engine): בשל מורכבות החישובים והעיבודים הרב-מימדיים, ניתן לפתח מנוע רב-מימדי מיוחד הפועל בשרת. קיימות שתי חלופות עיקריות למימוש המנוע הרב-מימדי - מנוע העובד עם נתונים המנוהלים במבנה רב-מימדי ייעודי, או לחילופין מנוע העובד באופן ישיר עם בסיסי נתונים טבלאיים. עבור מערכות המנהלות את הנתונים בבסיס נתונים רב-מימדי מיוחד המנוהל בשרת, המקום הטבעי לבצע את כל החישובים והעיבודים הוא, כמובן, באותו שרת. הקרבה בין הנתונים המנוהלים בשרת לבין החישובים המתבצעים בזיכרון של אותו שרת, יכולה להשפיע רבות על הביצועים הטובים שניתן לקבל במצב זה. במשך הזמן התפתחו גם מערכות המבוססות על מנוע רב-מימדי הפועל בשרת, ואילו הנתונים מנוהלים בבסיס נתונים טבלאי, המנוהל באותו שרת או בשרת אחר. מנוע זה משגר פקודות SQL לבסיס הנתונים, ומבצע את כל החישובים הדרושים. ברור, שלשיטה זו יש את יתרון העבודה הישירה מול הנתונים הפרטניים, אבל יש לו את החסרון של זמני תגובה איטיים יותר (מכיון שאינו מבצע חלק מהחישובים מראש).

- **במנוע בסיס הנתונים הטבלאי** (Relational Multi Dimensional Engine): החישובים והעיבודים יכולים להתבצע כחלק ממנוע בסיס הנתונים הטבלאי. יצרני בסיסי הנתונים הטבלאיים מנסים לשכלל את מנוע בסיס הנתונים ככל הניתן, כדי שמירב החישובים יבוצעו בצורה היעילה ביותר וכחלק ממנוע בסיס הנתונים. הבעיה היא ששפת SQL אינה תמיד השפה האידיאלית לביצוע חישובים אלה. היא אינה כזו מכיון שאינה תומכת באופן ישיר בחלק מהפעולות הנדרשות, ובדרך כלל צריכה לבצע מספר רב של מעברים על הנתונים (Multi Pass SQL) לביצוע החישובים.

- **במחשב המקומי** (Desktop Multi Dimensional Engine): בהתחשב בעובדה שהמחשבים האישיים הופכים למחשבים רבי עוצמה, ניתן לבצע חלק מהחישובים הנדרשים במחשב האישי עצמו, ולא בשרת. נדגיש רק שתפיסה זו מתנגשת, במידה מסוימת, במגמה העולמית של מעבר ל**מחשבי לקוח רזים** (Thin Client).

בהתבסס על החלופות השונות לאחסון הנתונים ולמקום ביצוע העיבודים, נקבל 9 ארכיטקטורות שונות של מערכות OLAP.

מיקום החישובים	מיקום הנתונים		
	בסיס נתונים טבלאי	בסיס נתונים רב מימדי	מבנה נתונים במחשב האישי
בשרת של בסיס הנתונים הרב מימדי		MOLAP	
בשרת של בסיס הנתונים הטבלאי	ROLAP		
במחשב האישי			Desktopl OLAP

תרשים 5.1: סוגי ארכיטקטורות למימוש מוצר OLAP.

מבין כל הארכיטקטורות האפשריות המוזכרות בתרשים 5.1, קיימות מספר ארכיטקטורות נפוצות יותר:

- ❖ **ארכיטקטורת MOLAP**: בה מנוהלים הנתונים בבסיס נתונים ייעודי בשרת, ובו גם מתבצעים רוב העיבודים. המוצרים הנפוצים ביותר בקטגוריה זו הם Express Server של חברת Oracle Corp, Holos של חברת Seagate, Essbase של חברת Hyperion Solutions Inc ואחרים.
- ❖ **ארכיטקטורת ROLAP**: בה מנוהלים הנתונים בבסיס נתונים טבלאי רגיל בשרת, והעיבודים מתבצעים במנוע בסיס הנתונים עצמו (או בשרת חישובים ייעודי). יש לשים לב לכך שמקובל לקרוא לארכיטקטורה זו ROLAP, גם אם העיבודים מתבצעים כחלק ממנוע בסיס הנתונים, כחלק ממנוע רב-מימדי נפרד מבסיס הנתונים או בשילוב מסוים ביניהם. רוב יצרני בסיסי הנתונים הטבלאיים נמצאים בשלב זה או אחר של פיתוח מנוע רב-מימדי, כחלק ממנוע בסיס הנתונים. המוצרים הנפוצים ביותר בקטגוריה זו הם MetaCube של חברת Informix Corp, DB/2 OLAP Server של חברת IBM, DSS Agent של חברת Micro Strategy, DecisionSuite של חברת Information Advantage, SQL Server OLAP Services של Microsoft.
- ❖ **ארכיטקטורת DOLAP** (Desktop OLAP) - בה מנוהלים הנתונים והחישובים במחשב האישי של המשתמש. ברור שארכיטקטורה זו מיועדת למשתמשים המבצעים ניתוחים רב-מימדיים פשוטים יחסית, בנפחי נתונים קטנים. בין כלים המבוססים על ארכיטקטורה זו נציין את PowerPlay של חברת Cognos, Business Objects של חברת Business Objects SA.
- ❖ **ארכיטקטורה מעורבת HOLAP** (Hybrid OLAP) - בנוסף לארכיטקטורות המוזכרות כאן ניתן גם לבנות ארכיטקטורה מעורבת, המשלבת את בסיס הנתונים הרב-מימדי הייעודי עם בסיס הנתונים הטבלאי, על ידי מתן אפשרות מעבר נוחה ביניהם. ארכיטקטורה זו מתאימה מאוד לארכיטקטורה רב-שכבתית של מחסן הנתונים, בה בסיס הנתונים הייעודי הוא חלק ממרכול הנתונים, ואילו בסיס הנתונים הטבלאי הוא חלק ממחסן הנתונים הארגוני. בין הכלים המבוססים על ארכיטקטורה זו נציין את Express של חברת Oracle, העושה מאמצים לשלב בין

בסיס הנתונים הייעודי לבין בסיס הנתונים הטבלאי של החברה, Media/MR של חברת Speedware ואת Hyperion Integration Server של חברת Hyperion, המאפשר למשתמשים מעבר קל מהמידע הסיכומי המנוהל ב-Essbase לנתונים פרטניים, המנוהלים בבסיס נתונים טבלאי ומוצרים אחרים.

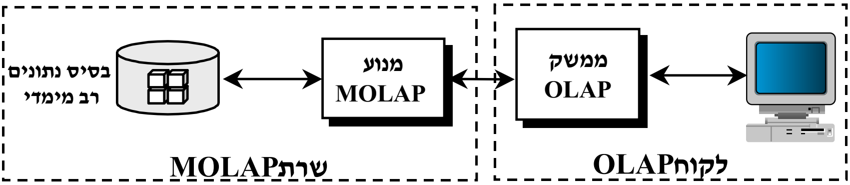
כעת, נסקור בצורה נרחבת יותר את הארכיטקטורות העיקריות של מוצרי OLAP.

מוצרים המבוססים על בסיסי נתונים רב-מימדיים (MOLAP - Multi dimensional OLAP)

תפיסת **ניתוח הנתונים הרב-מימדי** פותחה לפני מספר רב של שנים (תחילת שנות ה- 70). באותה תקופה היו בסיסי הנתונים הטבלאיים חדשים יחסית, ובעלי יכולות מוגבלות. מכיון שהתפיסה הטבלאית מבוססת על שורות ועמודות בטבלאות דו-מימדיות פשוטות, יש לבצע פעולות Join מורכבות למדי כדי לקבל את המבט הרב-מימדי. מסיבות אלה החליטו מספר חברות לפתח את התפיסה הרב-מימדית בהתבסס על בסיס נתונים ייעודי מיוחד - **בסיס הנתונים הרב-מימדי** (MDB - Multi Dimensional Database). עם חלוף השנים החלו לקרוא לארכיטקטורה זו בשם **MOLAP** - Multi Dimensional OLAP. שם ארכיטקטורה זו מטעה, מכיון שזהו שם כוללני לכל מוצר המאפשר ניתוח רב-מימדי. השם הנכון צריך להיות MDB-OLAP, אולם מכיון שהוא אינו נפוץ, נשתמש ב-MOLAP.

בסיס נתונים זה ניצל את המבנה הרב-מימדי באופן מיוחד, כדי לקבל זמני תגובה אופטימליים. מוצרים אלה מנהלים את הנתונים במבנה מיוחד של מעין מטריצות דלות (Sparse Matrix), הלוקחות בחשבון שלא בכל הצטלבות ערכי המימדים אכן קיימות עובדות אותן יש לנהל. יצרני המוצרים הרב-מימדיים פיתחו שיטות מיוחדות לניהול יעיל של קוביות דלות ערכים.

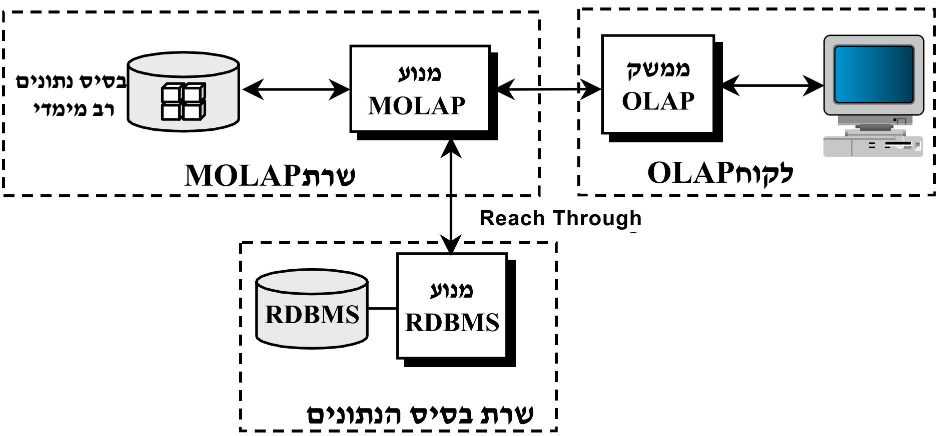
מכיון שנושא הסיכומים נפוץ מאוד בניתוח רב-מימדי, בחרו יצרנים אלה לבנות מראש מספר רב של סיכומים, בחתכים שונים, כך שתוך כדי ביצוע השאילתה לא יהיה צורך לבצע את הסיכומים. טכניקות אלה הביאו לזמני תגובה טובים מאוד של מוצרים אלה. זמני תגובה אלה הושגו לא מעט על חשבון זמן הטעינה של הקוביה, הזמן בו מבצעת המערכת את החישובים השונים, ומכינה את האינדקסים המיוחדים שלה. תיאור המנגנונים הפנימיים של מערכות המבוססות על בסיסי נתונים רב-מימדיים הם מעבר למטרות ספר זה.



תרשים 5.2: ארכיטקטורה עקרונית של מערכת MOLAP.

הארכיטקטורה המוצגת בתרשים 5.2 היא ארכיטקטורה טיפוסית של מערכת MOLAP המבוססת על שתי שכבות (Two Tier Architecture) - **שכבת שרת הנתונים**, המנהלת את בסיס הנתונים הרב-מימדי ואת המנוע המנהל את בסיס נתונים זה, **ושכבת הלקוח**, המכילה ממשק אל המנוע ואת אוסף כלי הצגת המידע.

מוצרים אלה (כגון Oracle Express, Hyperion Essbase ואחרים) מבוססים על כלי ניתוח רב-מימדי, ועל שרת רב-מימדי המנהל את בסיס הנתונים הרב-מימדי. בסיס הנתונים הרב-מימדי נטען מעת לעת ממחסן הנתונים, או לפעמים ישירות ממערכות תפעוליות (באותם ארגונים בהם מחסן נתונים לא קיים). מבחינת ארכיטקטורת מחסן הנתונים, ניתן להסתכל על בסיס הנתונים הרב-מימדי כחלק ממרכול נתונים (Data Mart) הנטען בנתונים ממחסן הנתונים. קיבלנו כאן את הארכיטקטורה הרב-שכבתית של מחסן הנתונים. במשך השנים הוסיפו יצרני הכלים הרב-מימדיים שכלולים רבים. אחת האפשרויות שהתווספה היתה היכולת לבצע שאילתות רב-מימדיות מול בסיס הנתונים הרב-מימדי, ותוך כדי תהליכי הקידוח להפעיל באופן אוטומטי שאילתות SQL ישירות מול בסיס הנתונים הטבלאי של מחסן הנתונים, לשם קבלת המידע הפרטני. לתכונה זו מקובל לקרוא SQL Pass Through או Reach Through.



תרשים 5.3: שילוב ארכיטקטורת MOLAP עם בסיס נתונים טבלאי.

בסיסי נתונים טבלאיים בעיצוב רב-מימדי (Relational Dimensional Database)

עם השיפורים שחלו בביצועים של מערכות לניהול בסיסי הנתונים הטבלאיים (RDBMS - Relational Data Base Systems) המסחריים ועם הפופולריות ההולכת וגוברת של מחסן הנתונים בכלל ותפיסת ניתוח המידע הרב-מימדית בפרט, התברר שניתן לבצע ניתוחים רב-מימדיים גם על בסיסי נתונים טבלאיים. זאת, בתנאי שהם יעוצבו בצורה מיוחדת - העיצוב הרב-מימדי של בסיס הנתונים הטבלאי.

טכניקות עיצוב הנתונים הרגילות, המבוססות על נירמול נתונים (Data Normalization), ושיטות ניתוח של יישויות וקשרים (ERD - Entity Relationship Diagrams), אינן מתאימות לניתוח הרב-מימדי הנדרש על ידי מקבלי ההחלטות. דבר זה נכון במיוחד אם ארגון מבקש לבנות קשרים מורכבים בין מספר רב של יישויות עסקיות. לדוגמה, חברה המבקשת להבין את הקשרים בין עלויות הייצור, גודל המוצר ומספר יחידות המוצר באריזה, תתקשה מאוד לעשות זאת באמצעות מודל נתונים טבלאי רגיל. הדרך הרגילה של בניית אוסף טבלאות מיוחדות לכל יישות, ואחר כך שימוש בפעולות Join מורכבות, היא דרך מורכבת מאוד. בדרך זו יהיה קשה מאוד לבצע ניתוח מגמות, השוואות בין מוצרים שונים וכדומה. התברר שיש לגשת לבעיה זו בדרך שונה לחלוטין מבחינת עיצוב בסיס הנתונים, **גישת העיצוב הרב-מימדית**. ההבדל בין שתי תפיסות עיצוב אלה (תפיסה רגילה של נירמול נתונים לעומת תפיסה של עיצוב רב-מימדי) הוא מנקודת מבטו של מנתח המידע ומקבל ההחלטות, כמו ההבדל בין להתמקד בבעיות המחשב לעומת התמקדות בבעיות העסקיות של הארגון.

תפיסת ה**עיצוב הרב-מימדית** (Relational Dimensional Database Design) של בסיסי נתונים טבלאיים פותחה במשך השנים על ידי גורמים שונים, אולם ללא ספק Ralph Kimball, שכתב את הספר המצוין "The Data Warehouse Toolkit" וכן בסדרה ארוכה של מאמרים בעיתון המקצועי "DBMS", הוא זה שהביא תפיסה זו לכדי תפיסת עיצוב שלמה, בזכות עצמה. בעקבותיו נכתבו מספר רב של ספרים ומאמרים, אולם זכות הראשונים היא לו. כיום, רוב מחסני הנתונים המודרניים מבוססים על תפיסת העיצוב של Ralph Kimball.

מעבר לתפיסה הרב-מימדית של העיצוב הלוגי, החלו החברות המייצרות מערכות RDBMS להוסיף תכונות מיוחדות, המיועדות בעיקר עבור מחסני נתונים. בין תוספות אלו ניתן לציין שיפורים מיוחדים ברכיב ה**אופטימיזציה** (Optimizer), המזהה את המבנה המיוחד של העיצוב הרב-מימדי, הכנסת מימד **מקביליות** (Parallelism) כמעט לכל רכיבי בסיס הנתונים כדי לאפשר ניצול טוב יותר של החומרה להשגת ביצועים טובים, שיפורים מיוחדים ב**רכיב הטעינה** (Loader), הוספת אפשרויות לניהול חכם של מחיצות (Partitions), טבלאות ועוד.

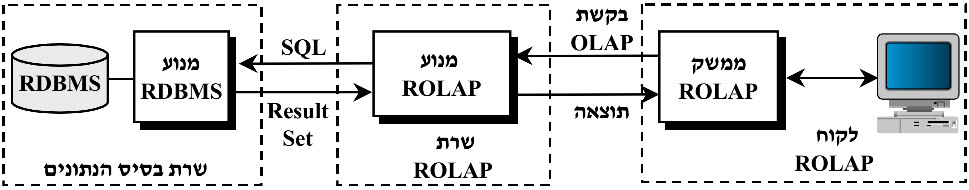
למרות העיצוב הרב-מימדי של בסיס הנתונים הטבלאי, שפת SQL אינה שפה מספיק עשירה וגמישה כדי לבצע את כל הפעולות והחישובים, המהווים חלק בלתי נפרד מהניתוח הרב-מימדי. העיצוב הרב-מימדי של בסיס הנתונים הטבלאי הוא רק ההתחלה של הניתוח הרב-מימדי וכדי לספק את מירב הפונקציונליות הנדרשת היה צורך להשלים את העיצוב הרב-מימדי בכלים ייעודיים המתמחים בביצוע כל החישובים והפעולות המקובלות בניתוח הרב-מימדי. זהו מקור התפתחותה של טכנולוגיית ROLAP, טכנולוגיה המשלימה את בסיס הנתונים הטבלאי המעוצב בצורה מיוחדת, במיגוון רחב של פונקציות תומכות. בכל מקרה, כל כלי ROLAP יוצאים מההנחה שבסיס הנתונים מעוצב בצורה רב-מימדית. בשל החשיבות ורוחב היריעה של שיטת עיצוב זו, יוקדש לה כל הפרק הבא.

מוצרים לניתוח רב-מימדי מבוסס בסיס נתונים טבלאי (ROLAP - Relational OLAP)

כאמור, בניית בסיס נתונים טבלאי בעיצוב רב-מימדי מהווה רק את הצעד הראשון לקראת אפשרות ניתוח רב-מימדי של מידע. במקביל לשינויים בעיצוב בסיס הנתונים נוצר הצורך לשפר את כלי הגישה והניתוח, העובדים עם בסיס הנתונים הטבלאי שעוצב בתפיסה הרב-מימדית. כפי שהוסבר, ניתוח רב-מימדי הוא הרבה יותר מאשר רק מבט רב-מימדי על הנתונים, והוא כולל עיבודים וחישובים מורכבים מאוד. גישה המבוססת על פעולות SQL בלבד היא עדיין מורכבת מדי, ושפת SQL חסרה מספר פונקציות חשובות מאוד מבחינת מנתח המידע.

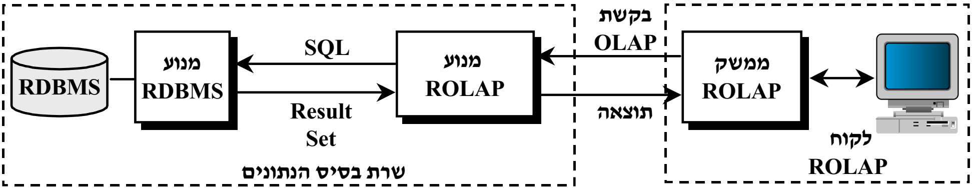
טכנולוגיית ROLAP מבוססת על בניית מנוע מיוחד, הנקרא בדרך כלל ROLAP Engine, שהוא שכבת תוכנה מיוחדת היודעת לבצע ולתמוך בכל הפעולות המיוחדות של הניתוח הרב-מימדי. שכבה זו מבצעת חלק גדול מהחישובים, ומשלימה את יכולות בסיסי הנתונים בכל הקשור לניתוח רב-מימדי. המנוע הרב-מימדי מבוסס על מילון נתונים (Meta Data Driven), המקבל את בקשות המשתמש, מתרגם אותן לפניות SQL חכמות לבסיס הנתונים, מנהל באופן אוטומטי טבלאות סיכום שונות, מבצע חישובים מיוחדים ולבסוף מחזיר את התוצאה למשתמש. המנוע מבצע כל זאת בזמן הריצה. במשך השנים נוצרו מספר ארכיטקטורות מימוש שונות של המנוע הרב-מימדי:

- ארכיטקטורת ROLAP בה המנוע הרב-מימדי הוא שכבת תוכנה נפרדת ממנוע בסיס הנתונים הטבלאי. עקרונית, ניתן לממש את המנוע הרב-מימדי הן בשרת מרכזי והן במחשב הלקוח. ארכיטקטורת ROLAP הפועלת בשרתים מיושמת על ידי יצרני צד שלישי, כגון DecisionSuite של חברת Information Advantage, DSS Agent של חברת Micro Strategy ואחרים. מוצרים אלה משלימים את בסיסי הנתונים הטבלאיים ביכולות ניתוח רב-מימדיות מיוחדות.



תרשים 5.4: ארכיטקטורת ROLAP עם מנוע ייעודי.

❖ ארכיטקטורת ROLAP בה המנוע הרב-מימדי הוא חלק ממנוע בסיס הנתונים הטבלאי. ארכיטקטורה זו מיושמת על ידי יצרני מערכות RDBMS, השוקדים כל הזמן על אינטגרציה הולכת וגדלה של יכולות OLAP בתוך בסיסי הנתונים שלהם. חלק מיצרנים אלה רכשו חברות המתמחות בנושא OLAP, ומשלבות טכנולוגיות אלו באופן מתמיד למוצרים העיקריים שלהם. ניתן להבחין במגמה זו כשלומדים על רכישתה של חברת IRI (יצרנית Express) על ידי Oracle, רכישת Stanford Technology Group (יצרנית Meta Cube) על ידי חברת Informix, רכישת IQ על ידי חברת Sybase, רכישת חברת Panorama הישראלית על ידי Microsoft, שילוב הטכנולוגיה של Essbase כחלק מבסיס הנתונים DB2 של חברת IBM, ועוד.



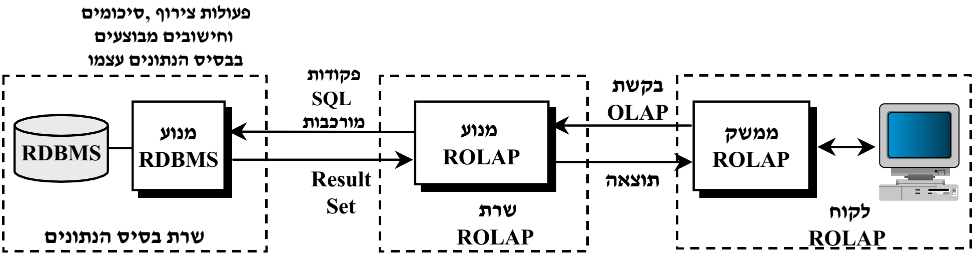
תרשים 5.5: ארכיטקטורת ROLAP המהווה הרחבה של מנוע בסיס הנתונים הטבלאי.

ללא קשר לצורת המימוש של המנוע הרב-מימדי, הנמצאת בתוך או ממעל בסיס הנתונים, ניתן להתייחס לארכיטקטורה העקרונית של מערכת ROLAP כאל **ארכיטקטורה תלת שכבתית** (Three Tier Architecture) - **שכבת בסיס הנתונים הטבלאי**, המבוסס על עיצוב רב-מימדי, **שכבת מנוע ROLAP**, המבצע את כל החישובים ומשמש כממשק גישה לבסיס הנתונים הטבלאי ו**שכבת הלקוח**, המכילה את הממשק למנוע ואת כלי הצגת המידע.

ללא קשר לצורת מימוש המנוע הרב-מימדי, טכנולוגיית ROLAP מאפשרת את הניתוח הרב-מימדי, ללא צורך בניהול הנתונים בבסיס נתונים רב-מימדי מיוחד. טכנולוגיה זו אינה טכנולוגיה נפרדת, העומדת בזכות עצמה, אלא מהווה נדבך נוסף מעל בסיס הנתונים הטבלאי, ומשלימה אותו במספר יכולות הקשורות לניתוח המידע הרב-מימדי. במובן מסוים, ניתן להתייחס לטכנולוגיה זו כאל טכנולוגיה משלימה לטכנולוגיית בסיסי הנתונים הטבלאיים. קיימות שיטות שונות למימוש מנוע ROLAP, לכל שיטה היתרונות והחסרונות שלה. נסקור כאן בקצרה את השיטות הנפוצות.

ארכיטקטורת SQL Cube

בארכיטקטורה זו נשענות רוב היכולות הרב-מימדיות של המנוע על יכולות בסיס הנתונים. המנוע מייצר פקודות SQL מורכבות, כדי לגשת אל הטבלאות הרגילות והטבלאות הסיכומיות בבסיס הנתונים, ומבצע את כל ה-Joins, הסיכומים והחישובים הרב-מימדיים במנוע בסיסי הנתונים הטבלאי. תוך כדי התהליך בונה המנוע הרב-מימדי מספר רב של טבלאות ביניים, לקבלת המבט הרב-מימדי.

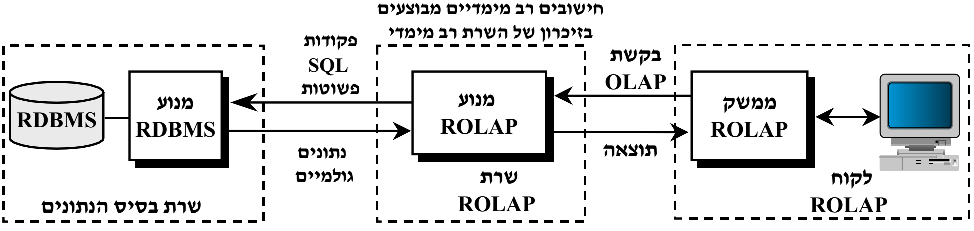


תרשים 5.6: מבנה עקרוני של ארכיטקטורת SQL Cube.

התלות המוחלטת של ארכיטקטורה זו ביכולות מערכת RDBMS יוצרת מספר בעיות קשות. כל החישובים הרב-מימדיים מוגבלים רק לאלה הנתמכים באופן ישיר על ידי בסיס הנתונים. חישובים מורכבים יותר, כגון ממוצעים משוקללים, השוואות רב-תקופתיות וכדומה, מורכבים מאוד למימוש באמצעות פקודות SQL וטבלאות ביניים. בעיה נוספת בארכיטקטורה זו היא יכולת מוגבלת מאוד בתמיכה במספר רב של משתמשים. התקורה הנוצרת כתוצאה מבניית טבלאות הביניים גדולה מאוד, וזמני התגובה נוטים להידרדר מהר מאוד עם הוספת משתמשים נוספים. רוב מוצרי ROLAP מהדור הראשון היו מבוססים על ארכיטקטורה זו.

ארכיטקטורת Hyper Cube

ארכיטקטורה זו מבוססת על מנוע רב-מימדי ייעודי, הפועל בנפרד ממנוע בסיס הנתונים הטבלאי. המנוע הרב-מימדי יכול לפעול בשרת נפרד, או בשרת בסיס הנתונים עצמו. מנוע זה מייצר פקודות SQL פשוטות, כדי לבנות ולשמור קוביה רב-מימדית בזיכרון השרת. ניתן להתייחס אל פקודות SQL כאל פקודות שמטרתן לבצע טעינה של נתונים לזיכרון.



תרשים 5.7: מבנה עקרוני של ארכיטקטורת Hyper Cube.

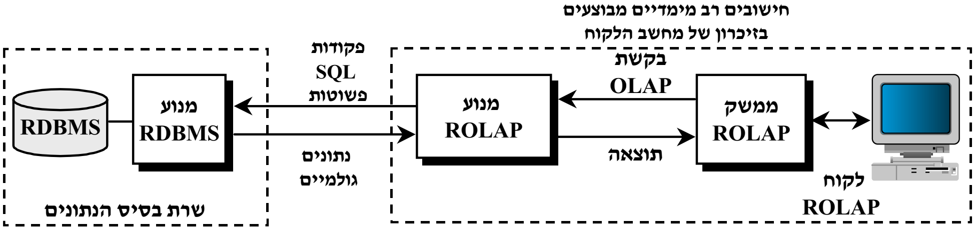
עם גמר בניית הקוביה הרב-מימדית בזיכרון, היא נשמרת שם וכל גישה נוספת של המשתמש היא אל הקוביה, ולא אל בסיס הנתונים עצמו. היתרון העיקרי של ארכיטקטורה זו הוא ביכולת לתמוך בחישובים רב-מימדיים מתוחכמים, מכיון שהקוביה מנוהלת בזיכרון על ידי מנוע רב-מימדי ייעודי, ואינה מוגבלת על ידי אילוצי מערכת RDBMS.

החסרונות העיקריים של ארכיטקטורה זו הם במגבלות ניהול קוביות רב-מימדיות גדולות מאוד, בשל מגבלות הזיכרון של השרת וזמן התגובה הרב שנמשכת הבנייה הראשונית של הקוביה בזיכרון. לאחר שהקוביה כבר מוכנה בזיכרון, משתפרים זמני התגובה, אבל הבנייה הראשונית יכולה להימשך זמן רב. הקוביה עצמה נשמרת רק לאורך השיח (Session) עם המשתמש, ולאחר מכן נמחקת. פנייה נוספת אל הקוביה בתום השיח מחייבת את בנייתה מחדש, דבר שכאמור יכול לארוך זמן רב. תפיסה זו אינה מתאימה למספר רב של משתמשים, מכיון שבשל מגבלת הזיכרון לא ניתן לבנות מספר גדול של קוביות פעילות בו-זמנית.

ארכיטקטורת Desktop Hyper Cube

ארכיטקטורה זו דומה לארכיטקטורה הקודמת של בניית קוביה רב-מימדית בזיכרון, אלא שהפעם הקוביה איננה נבנית בזיכרון השרת, אלא בזכירון של מחשב הלקוח. לאחר בניית הקוביה במחשב הלקוח, ניתן להתנתק מהשרת ולבצע סדרה שלמה של ניתוחים, מבלי ליצור עומס בשרת. המגבלות של שיטה זו הן מגבלות הזיכרון של מחשב הלקוח. בשל הצורך להעביר כמויות גדולות של נתונים משרת בסיס הנתונים הטבלאי אל מחשב הלקוח דרך רשת התקשורת, נוצר עיכוב נוסף בזמן התגובה. שיטה זו טובה לאותם מקרים בהם הקוביה קטנה ואין צורך בשיתוף הקוביה בין מספר מנתחי מידע - כל מנתח עובד על הקוביה שלו בלבד.

מוצרים אלה הם בדרך כלל זולים יותר מהמוצרים מבוססי שרת.

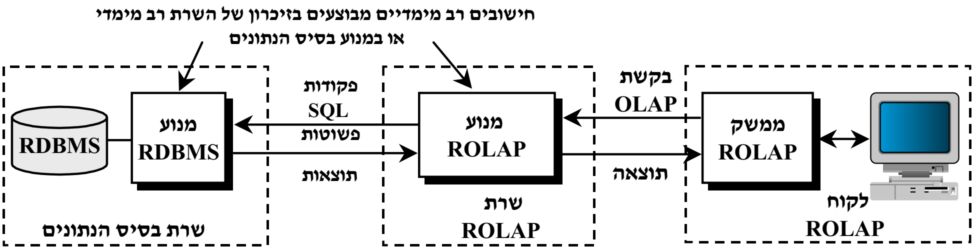


תרשים 5.8: ארכיטקטורת Desktop Hyper Cube.

ארכיטקטורת Flash Cube

ארכיטקטורה זו מהווה שילוב מסוים של שתי הארכיטקטורות הקודמות - ארכיטקטורת SQL Cube, המטילה את רוב החישובים הרב-מימדיים על בסיס הנתונים הטבלאי, וארכיטקטורת Hyper Cube, המטילה את רוב החישובים על מנוע רב-מימדי נפרד. ארכיטקטורת Flash Cube דומה לארכיטקטורת Hyper Cube, אבל המנוע שלה עושה שימוש נרחב יותר ביכולות מערכת RDBMS. בדרך זו, החישובים הרב-מימדיים מבוזרים בין מנוע בסיס הנתונים הטבלאי לבין המנוע הרב-מימדי. גם בארכיטקטורה זו ניתן להפעיל מנוע הרב-מימדי בשרת נפרד או באותו שרת, כמו מערכת RDBMS. בהתבסס על חוקים הנשמרים במילון נתונים, מקבל המנוע הרב-מימדי החלטה היכן לבצע את הפעולות הרב-מימדיות - בבסיס הנתונים הטבלאי או במנוע הרב-מימדי הייעודי. המנוע הרב-מימדי מכיר את מבנה בסיס הנתונים ואת הטבלאות שלו, טבלאות רגילות וטבלאות סיכומיות, ועושה שימוש בידע זה כדי להחליט לאיזו טבלה לפנות, איזו פקודת SQL לייצר ואילו עיבודים נוספים לבצע בשרת הרב-מימדי.

בדומה לארכיטקטורת SQL Cube, גם בארכיטקטורה זו מתבצעים חלק מהעיבודים בבסיס הנתונים הטבלאי. כתוצאה מכך יש פחות עומס על הרשת, בשל הצורך להעביר נתונים מבסיס הנתונים החוצה, אל הקוביה בזיכרון (של השרת או של הלקוח). מכיון שחלק מהפעולות מתבצעות ישירות במערכת RDBMS, ניתן לטפל גם בטבלאות גדולות מאוד. עם זאת, תפיסה זו אינה מגבילה את עצמה רק ליכולות הקיימות במערכת RDBMS בלבד. היא משלימה אותו בסדרה שלמה של פעולות רב-מימדיות המתבצעות במנוע הרב-מימדי. ארכיטקטורה זו יכולה לתמוך במספר רב של משתמשים בו-זמנית.



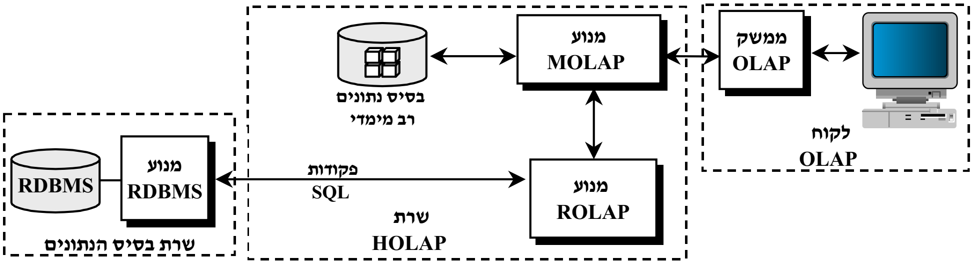
תרשים 5.9: מבנה עקרוני של מערכת מבוססת Flash Cube.

ארכיטקטורה רב-מימדית מעורבת (Hybrid OLAP)

במשך תקופה מסוימת התפתחו מאבקים שיווקיים בין יצרני כלי התוכנה מבוססי טכנולוגיית MOLAP, לבין יצרני מערכות RDBMS, שהוסיפו ושיכללו את המוצרים שלהם לקבלת יכולות OLAP בבסיס נתונים טבלאי רגיל ולבין יצרני מערכות ROLAP שהשלימו את יכולות בסיס הנתונים הטבלאי, על ידי הוספת מנוע ייעודי חיצוני.

בשנים האחרונות נחלש מאבק זה וכמעט נעלם, הן מכיון שהגבולות הפכו לעמומים והן בשל ההבנה שלכל ארכיטקטורה יש יתרונות מסוימים, וכי הדרך הנכונה היא לשלב ביניהן.

התברר, שבאופן מעשי אלה שתי תפיסות משלימות, ולא מתחרות. במחסן הנתונים המודרני יש מקום לשתי התפיסות. בשל נפחי הנתונים הגדולים של מחסן הנתונים, טכנולוגיית ROLAP מתאימה מאוד. התאמה זו נובעת מכך שטכנולוגיית ROLAP מאפשרת ניתוח מידע רב-מימדי, באופן ישיר מעל בסיס הנתונים של מחסן הנתונים, ללא צורך בגזירות והעברות נתונים נוספות. מצד שני טכנולוגיית MOLAP, המצטיינת בביצועים מעולים וכלי ניתוח מתקדמים יותר, הפכה לטכנולוגיה המתאימה ביותר לניהול מרכול נתונים. הנוחות במעבר המהיר מבסיס הנתונים הרב-מימדי לבסיס הנתונים הטבלאי, העומד בלב מחסן הנתונים, רק חיזקה את התפיסה שאלו הן תפיסות משלימות, לא מתחרות.



תרשים 5.10: שילוב בסיס נתונים רב-מימדי עם בסיס נתונים טבלאי.

הארכיטקטורה המשולבת מנסה ליהנות מהיתרונות בשני העולמות גם יחד - העוצמה, העושר הפונקציונלי, מהירות התגובה של תפיסת MOLAP בכל הקשור לנתונים ברמה סיכומית יחד עם האמינות, היכולת לנהל את הרמה הפרטנית של הנתונים של תפיסת ROLAP וגם כושר הגידול. לארכיטקטורה זו מקובל כיום לקרוא **HOLAP**. HOLAP מאפשרת מידה רבה של שקיפות, מבחינת המשתמש, בגישה הן לרמות הסיכומיות (המנוהלות בבסיס נתונים רב-מימדי ייעודי), והן לרמות הפרטניות (המנוהלות בבסיס הנתונים הטבלאי).

בנוסף ליצרני מערכות RDBMS, המרחיבים כל הזמן את היכולות הרב-מימדיות של בסיסי הנתונים שלהם, הופיעו יצרנים נוספים המציעים מוצרי HOLAP. ביניהם ניתן לציין את המוצר Media/MR של חברת Speedware. כל המוצרים האלה יודעים לעשות שימוש במידע המנוהל בקטלוג של בסיס הנתונים הטבלאי (Meta Data), ללא צורך בניהול ותחזוקה כפולה של מידע זה. מוצרים אלה יודעים לחולל פקודות SQL הנבנות באופן אוטומטי, וללא ידיעת המשתמש.

סקירת מוצרי OLAP
על פי מטריצת הארכיטקטורות

לאחר שסקרנו את הארכיטקטורות השונות למימוש מוצרי OLAP, נסקור את המוצרים הנפוצים בתחום זה, על פי המטריצה שהוצגה בסעיף הקודם.

	מיקום הנתונים		
מיקום החישובים	**בסיס נתונים טבלאי**	**בסיס נתונים רב מימדי**	**מבנה נתונים במחשב האישי**
במנוע רב מימדי בשרת	**Micro Strategy DSS Agent** **Information Advantage DecisionSuite** **IBM DB2 OLAP Server** **Informix MetaCube** **Segate Holos (ROLAP Mode)** **Oracle Express (ROLAP Mode)** **Microsoft OLAP Services** 1	**Hyperion Essbase** **Oracle Express** **Segate Holos** **Gentia** **SAS CFO Vision** **Comsare Decision** **Applix TM/1** **Pilot Analysis Server** 4	
במנוע בסיס הנתונים הטבלאי	**Orcacle** **Informix MetaCube Datablade** **IBM DB2** 2		
במנוע רב מימדי במחשב האישי	**Cognos PowerPlay (Server Option)** 3	**Hyperion Enterprise** **Hyperion Pillar** **Comshare FDC** 5	**BusinessObject** **Cognos PowerPlay** **Oracle Personal Express** **Brio Query** **TM/1 Perspectives** 6

תרשים 5.11: טבלת מוצרי OLAP לפי הארכיטקטורה.

להלן מספר הערות לטבלה שבתרשים 5.11:

- ❖ **משבצת 1**: מוצרים אלה ידועים בשם ROLAP. הם מבוססים על מנוע רב-מימדי מיוחד (הפועל בשרת), ליד או בנפרד ממערכת RDBMS. מבין מוצרים אלה, השניים הראשונים מבוססים רק על תפיסת ROLAP, בעוד שאחרים מבוססים על שילוב מסוים בין ROLAP ו-MOLAP.
- ❖ **משבצת 2**: למוצרים אלה מנוע רב-מימדי המהווה חלק ממנוע בסיס הנתונים עצמו ונמצאים בשלבי פיתוח כאלה ואחרים. חלק מהחברות כבר שחררו גרסאות ראשוניות, המכילות את מנוע העיבוד הרב-מימדי כחלק ממנוע בסיס הנתונים. למשל, חברת Informix ששחררה גירסה של Informix Universal Server, המכיל אופציה של מנוע רב-מימדי פנימי. גם חברת יבמ שחררה לשוק מוצר דומה במערכת DB2 שהיא משווקת.
- ❖ **משבצת 4**: אלה מוצרי MOLAP המובילים. הם מבוססים על מנוע רב-מימדי ובסיס נתונים רב-מימדי, הפועלים בשרת.
- ❖ **משבצת 6**: אלה המוצרים המובילים לניתוח רב-מימדי, המבוססים על תחנת העבודה האישית של המשתמש.

ראוי להדגיש שקיימים הבדלים מהותיים בין המוצרים השונים, אפילו אם הם מופיעים באותה משבצת. כל אחד מהמוצרים נבדל בעוצמת המנוע הרב-מימדי, ברמת תחכום האופרטורים המיוחדים בהם הוא תומך, בידידותיות למשתמש, בביצועים, בנפח הנתונים שהוא מסוגל לנהל וביכולת הגידול שלהם. מסיבה זו סביר להניח שנמצא באותו מחסן נתונים מספר מוצרים שונים, כל אחד עבור המשימות אותן הוא מבצע טוב יותר.

מה הארכיטקטורה המומלצת?

אין תשובה אחת לשאלה זו, אולם ברור שכיום שאלה זו הופכת לפחות ופחות רלוונטית, בשל השילוב שנוצר בין הארכיטקטורות השונות. יחד עם זאת, קיימים מספר קריטריונים עליהם יש לתת את הדעת בעת ההחלטה על המוצר והארכיטקטורה המתאימה ביותר לארגון.

❖ **גודל בסיס הנתונים** (Database Size): נכון להיום, טכנולוגיית בסיסי הנתונים הייעודיים בארכיטקטורת MOLAP מסוגלת להתמודד עם נפח נתונים בתחום של 10 ועד 20 ג'יגה-בית (GB). למרות שאלה נפחים מכובדים למדי, סדר הגודל של מחסני הנתונים המודרניים כיום נמנה במאות ג'יגה-בית ויותר. מכאן, ברור שארכיטקטורת MOLAP תתאים ביותר לניהול מרכול הנתונים, שהוא בדרך כלל מצומצם יותר בנפחו, מכיון שהוא מכיל בעיקר מידע סיכומי. כשצריך לבצע ניתוחי מידע ברמה פרטנית יותר על נפחי נתונים גדולים, הארכיטקטורה המתאימה ביותר היא ארכיטקטורת ROLAP.

❖ **קצב השינויים** (Source Data Volatility): אם הנתונים המשמשים את תהליכי ניתוח הנתונים משתנים בתדירות גבוהה, נוצרת בעיה בארכיטקטורת MOLAP. ארכיטקטורה זו מבוססת על בסיס נתונים נפרד, ולכן יש לבצע את טעינתו מחדש בתדירות גבוהה. תהליך טעינה זה יכול להימשך זמן רב ולצרוך משאבי מחשוב רבים. ארכיטקטורת ROLAP, מכיון שהיא עובדת ישירות מעל מערכת RDBMS, אינה סובלת מבעיה זו.

❖ **ביצועים** (Performance): ארכיטקטורת MOLAP נהנית מביצועים טובים יותר. זאת מכיון שחלק גדול מאוד מהחישובים והסיכומים מתבצע תוך כדי תהליך הטעינה. לעומתה, ארכיטקטורת ROLAP, למרות שגם היא מבוססת על ביצוע סיכומים תוך כדי טעינה, עדיין אינה מגיעה לרמת הביצועים של בסיסי הנתונים הרב-מימדיים הייעודיים, אם כי הפער הולך ומצטמצם כל הזמן.

❖ **פונקציונליות רב-מימדית** (Multi Dimensional Functionality): ארכיטקטורת MOLAP מכילה עושר פונקציונלי רב יותר מאשר ארכיטקטורת ROLAP, אם כי הפערים הולכים ומצטמצמים. צריך לזכור שניתוח רב-מימדי הוא הרבה יותר מאשר הצגת הנתונים במבנה רב-מימדי. הוא דורש תמיכה באופרטורים מיוחדים ובביצוע חישובים מורכבים המהווים חלק מתהליך בניית המודלים. חלק מפונקציונליות זו מזכירה את הגיליונות האלקטרוניים ואת יכולותיהם להגדיר נוסחאות מורכבות, אריתמטיקה של מטריצות, ביצוע תהליכי סיכומים מהירים וכדומה.

- **רמות הסיכומים (Levels of Aggregation)**: ככל שרמת הסיכומים השונים הנדרשת גבוהה יותר, ארכיטקטורת MOLAP מתאימה יותר. התאימות הגבוהה נובעת מכך שממילא מתבצעים מספר רב מאוד של סיכומים תוך כדי הטעינה. לעומת זאת, ניהול הסיכומים בארכיטקטורת ROLAP מורכב יותר, ובשלב זה עדיין אינו אוטומטי. דבר זה מחייב התערבות של מנהלי בסיס הנתונים בבניית הסיכומים. בנוסף, רוב כלי השאילתות עדיין אינם יודעים לנצל באופן אוטומטי את הטבלאות הסיכומיות.
- **בסיס נתונים נפרד (Separate Database)**: ניהול בסיס נתונים רב-מימדי ייעודי מוסיף רמה נוספת של סיבוכיות תפעולית. צריך להכיר את כלי הניהול של בסיס הנתונים הרב-מימדי, לדאוג לכוונון שלו, לעדכן אותו מעת לעת, לוודא שהוא שותף בתהליכי גיבוי ושחזור וכדומה. מכיון שזהו רובד נוסף במחסן הנתונים, באופן טבעי קיימת כאן רמת סיבוכיות נוספת.
- **שפת גישה קניינית (Proprietary Data Access Language)**: רוב המערכות המבוססות על ארכיטקטורת MOLAP משתמשות בשפת גישה מיוחדת לנתונים. זו שפה שונה משפת SQL, ולחלקן יש ממשקי תכנות (API) מיוחדים. דבר זה מחייב לימוד שפת גישה נוספת, ובנוסף מאפשר רק למספר מוגבל של כלי צד שלישי גישה לבסיס הנתונים הרב-מימדי.

כללי Codd להגדרת מוצרים רב-מימדיים

בשנת 1993 פרסם **Codd** מאמר, שהוזמן על ידי חברת Arbor Software. המאמר הנושא את הכותרת "Providing OLAP to Users and Analysts: An IT Mandate" הציג אוסף של 12 כללים המגדירים את התכונות אותן חייב מוצר OLAP לקיים ובהן הוא חייב לתמוך. Codd הוא חוקר ומדען בעל מוניטין רבים בקרב אנשי טכנולוגיית המידע, כאשר עיקר פרסומו בא מתחום בסיסי הנתונים הטבלאיים ונחשב לממציא **מודל הנתונים הטבלאי**. למרות פרסומו הרב והמוניטין המקצועיים שלו, עורר מאמר זה ויכוחים רבים בקרב אנשי המקצוע, בעיקר בשל העובדה שהוא הוזמן על ידי חברה מסחרית. מחלוקת נוספת התפתחה מכיון שהכללים נוסחו באופן כוללני מדי, ונותר מקום רב לפרשנויות שונות. למרות המחלוקות שמאמר זה עורר, ולמרות שאין להתייחס למאמר זה כאל מאמר אקדמי מבוסס (אלא יותר כאל מסמך שפורסם על ידי גורם מסחרי), החשיבות שלו נובעת מהעובדה שהוא מגדיר מספר עקרונות בהם מוצרים המבקשים להיקרא **מוצרי OLAP** צריכים לתמוך ולקיים.

נציג כאן את 12 הכללים (למרות שבשנת 1995 הם הורחבו ל-18 כללים) במטרה לאפשר לקורא להתרשם מהם עקרונות מוצרי OLAP. הכרת כללים אלה יכולה לסייע בהבנת מוצרי OLAP ושיטות המימוש השונות שלהם. ניתן לומר, שרוב המוצרים המסחריים הקיימים כיום אינם תומכים באופן מלא **בכל** הכללים, אלא רק בחלקם.

- ❖ **כלל מספר 1 - מבט לוגי רב-מימדי** (Multi Dimensional Conceptual View) - זהו הכלל הבסיסי ביותר הקובע שעל המוצר לתמוך במבט הרב-מימדי על הנתונים, על כל המשתמע מכך. הפרק הקודם הסביר מהו המבט הרב-מימדי, לכן לא נחזור כאן על התכונות הנדרשות.

- ❖ **כלל מספר 2 - שקיפות** (Transparency) - כלל זה דורש שקיפות מירבית, מנקודת מבט המשתמש, לגבי מקור הנתונים. המשתמש יכול לפנות למנוע OLAP לקבלת נתונים תוך שהוא עובד עם ממשק של המוצר או ממשק אחר (כמו גיליון אלקטרוני). הוא אינו צריך להיות ער לכך שהנתונים מגיעים מבסיס הנתונים של המוצר, או מכל מקור אחר. כדי לתמוך בתכונת השקיפות, על המוצר לאפשר קשר מקוון חם עם מספר מקורות הטרוגניים של נתונים.

- ❖ **כלל מספר 3 - נגישות** (Accessibility) - כלל זה קובע שהמוצר צריך להתבסס על מנוע OLAP הנמצא בתווך, בין מקורות הנתונים לבין ממשק המשתמש.

- ❖ **כלל מספר 4 - ביצועי דיווחים אחידים** (Uniform Reporting Performance) - כלל זה קובע שביצועי המוצר חייבים להיות טובים ואחידים, גם אם מספר המימדים (או נפח בסיס הנתונים) גדל. כלל זה מכיל סתירה מסוימת והיא, שמצד אחד ככל שמספר הנתונים שיש להציג גדל, סביר להניח שזמן התגובה יהיה גדול יותר ואילו מצד שני, מכיון שמספר המימדים גדול יותר, לכן סביר להניח שנרצה להציג יותר נתונים. כלל זה נותן העדפה מסוימת למוצרי MOLAP, מכיון שהם מבצעים טעינה מראש של הנתונים. לכן, הקנס על ריבוי/הגדלת מספר המימדים יהיה בעיקר בזמן הטעינה, לאו דווקא בזמן עבודת המשתמש. מוצרי ROLAP, המבצעים חלק גדול יותר של החישובים בזמן אמיתי, סובלים מירידה בביצועים עם גידול מספר המימדים.

- ❖ **כלל מספר 5 - ארכיטקטורת שרת/לקוח** (Client/Server Architecture) - כלל זה קובע שהמוצר חייב להתבסס על ארכיטקטורת שרת/לקוח. עוד הוא קובע שרכיב השרת של המוצר חייב לאפשר ללקוחות שונים להתחבר אליו בצורה נוחה, קלה וללא מאמצי תכנות רבים.

- ❖ **כלל מספר 6 - רב-מימדיות כללית** (Generic Dimensionality) - כלל זה קובע שהתמיכה מבחינת המבנה, והפעולות בכל המימדים חייבים להיות זהים. יחד עם דרישה זו לאחידות הטיפול בכל המימדים, מאפשר הכלל לתמוך בפעולות מיוחדות למימדים ייחודיים (כמו למשל, מימד הזמן). כלל זה עורר ויכוחים רבים בין אנשי המקצוע, מכיון שהכלליות חשובה אולם גם ניצול תכונות ייחודיות של מימד חשובה, ויכולה להקל במידה רבה את הטיפול במימד.

- ❖ **כלל מספר 7 - התאמה אוטומטית של הרמה הפיסית** (Automatic Adjustment of Physical Level) - כלל זה דורש כי המוצר יעדכן באופן אוטומטי את הסכימה הפיסית שלו כתוצאה משינויים במודל, משינויים בנפחי הנתונים ומשינויים בדלילות הקוביה הרב-מימדית.

❖ **כלל מספר 8 - תמיכה בריבוי משתמשים** (Multi User Support) - כלל זה קובע שהמוצר צריך לתמוך במספר משתמשים הניגשים למודל בו-זמנית. הכלל מרחיב יכולת זו מעבר ליכולת קריאה בלבד (Read Only) של נתונים, מכיון שחלק מהניתוחים מחייבים עדכונים לצורך סימולציות וניתוחי What If.

❖ **כלל מספר 9 - פעולות חוצות מימדים ללא הגבלה** (Unrestricted Cross Dimensional Operations) - כלל זה קובע שכל החישובים והפעולות השונות צריכים להיתמך בכל המימדים.

❖ **כלל מספר 10 - טיפול אינטואיטיבי בנתונים** (Intuitive Data Manipulation) - כלל זה קובע שכל ממשק המשתמש חייב להיות ידידותי ונוח, תוך מתן אפשרות לבצע את מירב העבודה באופן ישיר בתאים המוצגים.

❖ **כלל מספר 11 - דיווח גמיש** (Flexible Reporting) - כלל זה קובע שהמוצר צריך לאפשר הצגת המימדים במיגוון גדול של אפשרויות, על פי צרכי המשתמש.

❖ **כלל מספר 12 - מספר מימדים ורמות סיכומיות בלתי מוגבל** (Unlimited Dimensions and Aggregation Levels) - כלל זה קובע שעל המוצר לתמוך במספר בלתי מוגבל של מימדים ורמות סיכום. במציאות, ברור שלא ניתן לקיים כלל זה, בשל מגבלות שונות כמו גודל זיכרון, עוצמת המחשב וכדומה. מוצר התומך ב-15 עד 20 מימדים צריך להספיק לכל צורך מעשי. לעיתים די נדירות נדרשים יותר מ-10 מימדים. לכן, גם תמיכה במספר קטן יותר של מימדים סבירה.

סיכום

בפרק זה סקרנו את הארכיטקטורות השונות של מוצרי OLAP. במשך השנים הופיעו ארכיטקטורות שונות, בעיקר בשל הפופולריות העצומה שקיבל נושא הניתוח הרב-מימדי. שתי הארכיטקטורות העיקריות למימוש ניתוח המידע הרב-מימדי הן - ארכיטקטורת MOLAP, המנהלת את הנתונים בבסיס נתונים רב-מימדי בעל מבנה מיוחד או ארכיטקטורת ROLAP, המוסיפה שכבה של מנוע רב-מימדי ליד (או אל תוך) בסיס הנתונים הטבלאי, המעוצב בצורה במבנה רב-מימדי.

שתי ארכיטקטורות אלו משלימות אחת את השנייה, וברוב המקרים נמצא את שתיהן מיושמות זו לצד זו במחסן הנתונים:

❖ ארכיטקטורת ROLAP, הפועלת ישירות מול בסיסי הנתונים הענקיים ונותנת יכולות ניתוח רב-מימדיות, אם כי לא תומכת באופן ישיר בכל עושר האופרטורים וסוגי הניתוחים שמספקים מוצרי MOLAP.

❖ ארכיטקטורת MOLAP המיושמת כחלק ממרכול נתונים, כאשר טעינת הנתונים מתבצעת מבסיס הנתונים הטבלאי של מחסן הנתונים. מוצרים אלה מאפשרים גישה ישירה מכלי הניתוח הרב-מימדיים אל בסיס הנתונים הטבלאי, לקבלת מידע ברמה פרטנית.

למרות שהיצרנים השונים היו רוצים שנחשוב שיש כאן שתי תפיסות עולם שונות באופן מהותי, מומלץ להימנע מלהיכנס למלחמה זו. רצוי להבין ששתי התפיסות חשובות וכי לשתיהן מקום במחסן הנתונים. הצורה הנכונה יותר להסתכל בטכנולוגיית כלי הניתוח הרב-מימדיים היא בהתאם לעוצמת היכולות האנליטיות של הכלים, ועל פי רמת תחכום הניתוחים להם זקוקים המשתמשים. בהקשר זה, מקובל לחלק את הכלים לשלוש קטגוריות:

- ❖ כלים המיועדים בעיקר להצגת נתונים בצורה רב-מימדית, ובעלי יכולות ניתוח מוגבלות.
- ❖ כלים המאפשרים, בנוסף, גם יכולות ניתוח מורכבות כגון ממוצעים נעים, הצגת Top 5 וכדומה.
- ❖ כלים המאפשרים, בנוסף, גם יכולות ניתוח מתקדמות, כגון אלגוריתמים לחיזוי, ניתוח תבניות, כללים להקצאה וכדומה.

מכיון שכל אחד מכלים אלה מתאים, בדרך כלל, לאוכלוסיית משתמשים שונה, מקובל מאוד למצוא באותו מחסן נתונים מספר כלים שונים, כל אחד עבור אוכלוסיית מטרה שונה.

פרק 6: עיצוב רב-מימדי של בסיסי נתונים טבלאיים (Relational Dimensional Database Design)

מבוא

בפרק זה נציג את העקרונות לעיצוב בסיס הנתונים הפועל בסביבת מחסן הנתונים. במשך שנים למדו ארגונים ואנשי תוכנה לבנות **מערכות מקוונות לעיבוד תנועות - OLTP** (On Line Transaction Processing) שמטרתן העיקרית היא עיבוד ואיסוף נתונים, ובסופו של דבר, אוטומציה של תהליכי העבודה החוזרים על עצמם מספר פעמים רב. מודל הנתונים נבנה במטרה לייעל ככל הניתן את תהליכי הזנת הנתונים (Data Entry), וקיים בו דגש קטן מאוד על אחזור הנתונים למטרות קבלת החלטות.

מטרותיו של מחסן הנתונים שונות מאוד. בעוד איסוף הנתונים ועיבודם מתבצע על ידי המערכות התפעוליות, תפקיד מחסן הנתונים הוא לתמוך בתהליכי הצגת וניתוח המידע לדרג מקבלי ההחלטות בארגון. יש לזכור שמודל הנתונים של מחסן הנתונים חשוף מאוד למשתמשי הקצה, והם פועלים על המודל באופן ישיר באמצעות מחוללי שאילתות ודוחות למיניהם. טכניקת העיצוב המוצגת בפרק זה היא כללית, וניתן ליישמה בכל בסיסי הנתונים הטבלאיים המסחריים הנפוצים, וביניהם Oracle, Informix, Sybase, DB2, RedBrick, Terradata, SQL Server ואחרים.

הטכניקה העיקרית בה נעשה שימוש היא טכניקת **עיצוב בסיסי נתונים רב-מימדיים** (Multi Dimensional Databases - MDB). זו טכניקת עיצוב מיוחדת שפותחה עבור סביבות תומכות החלטה. היא פותחה ושוכללה בעיקר על ידי Dr. Ralph Kimball, ומוצגת בספרו המצוין "The Data Warehouse Toolkit". חלק גדול מהרעיונות והדוגמאות המופיעות בפרק זה מבוססות על ספר זה. נדגיש, שטכניקת עיצוב זו מתאימה הן עבור מחסן הנתונים והן עבור מרכול הנתונים.

מדוע נדרשת טכניקת עיצוב מיוחדת

אחת השאלות הראשונות שעלינו לשאול היא מדוע צריך טכניקת עיצוב חדשה של מודל הנתונים. האם כל מה שלמדנו על עיצוב בסיסי נתונים בסביבות תפעוליות, לא יכול להתאים גם למחסן הנתונים? ובכן, מסתבר שבאופן חלקי מאוד. אין זה צריך להפתיע, בהתחשב בכך שמטרות עיצוב מודל הנתונים בסביבת מחסן נתונים שונות באופן מהותי ממטרות העיצוב בסביבה עתירת תנועות.

נתייחס למספר מאפיינים ועקרונות עיצוב, בהם יש מספר חסרונות בולטים לטכניקות העיצוב של בסיסי נתונים הפועלים בסביבות עיבוד תנועות, לעומת הצרכים של עיצוב בסיסי נתונים הפועלים בסביבת מחסן נתונים.

שיקולי עיצוב עיקריים בסביבת עיבוד תנועות (OLTP)

נסקור בקצרה את הדגשים והשיקולים העיקריים בעיצוב בסיס נתונים הפועל בסביבת עיבוד תנועות (OLTP). טכניקות עיצוב אלה התפתחו במשך מספר שנים, ומבוססות על **מודל יישויות-קשרים** (Entity Relationship Data Model), ועל טכניקת נירמול הנתונים (Data Normalization), שמטרתה העיקרית לצמצם עד למינימום את כפילות הנתונים.

זמן ביצוע תנועה (Transaction Performance)

טכניקות העיצוב של בסיסי הנתונים ב**סביבות עיבוד תנועות** ממוקדות מאוד בנושא ביצועי תנועות המעדכנות את בסיס הנתונים. מכיון שבסיסי הנתונים התפעוליים מעודכנים במהלך יום העבודה ממאות ואלפי תחנות קצה בו-זמנית, יש חשיבות מרובה להגיע לזמני עיבוד תנועה קצרים ביותר (שניות ספורות, לכל היותר). מכיון שרוב המערכות התפעוליות המודרניות הן מערכות מקוונות, המשרתות את העבודה השוטפת של הארגון (כמו ביצוע פעולה במכשיר אוטומטי של בנק, רישום קריאת שירות לתיקון מכשיר חשמלי, בירור מצב החשבון האחרון שנשלח ללקוח, רישום תנועה חדשה במערכת המלאי, שינוי סטטוס הזמנת רכש וכדומה), זמן התגובה הוא אחד הפרמטרים החשובים ביותר בהצלחת יישום המערכת.

במערכת תפעולית, בה אותה תנועה מבוצעת פעם אחר פעם, לעיתים מאות ואלפי פעמים באותו יום, אילוץ זמן התגובה הוא אילוץ מרכזי. לכן, מוקדש מאמץ ניכר להבטיח שעיבוד תנועה יסתיים בתוך שנייה, עד מספר בודד של שניות. אחת הסיבות שניתן לעמוד באילוץ זמני התגובה הקצרים מתבסס על העובדה שתנועות מקוונות מעדכנות, בדרך כלל, מספר קטן של טבלאות. בדרך כלל אמורים הדברים לגבי טבלה בודדת ועשויים, במקרים המורכבים יותר, להגיע עד כדי 10 טבלאות. רק במקרים נדירים צריך שאותה תנועה תעדכן מספר רב יותר של טבלאות. לעומת מצב זה, בסביבת מחסן הנתונים לא מתבצע עדכון, אלא רק אחזור נתונים. לכן, מיקוד מעצב בסיס הנתונים במקרה זה הוא להבטיח ששאילתות שכיחות תסתיימנה בפרקי זמן קצרים.

מורכבות אינה בעיה

מכיון שהגישה לנתונים במערכת תפעולית מתבצעת תמיד דרך יישומים הנכתבים על ידי מהנדסי תוכנה מקצועיים, יכול העיצוב המתקבל להיות מורכב מאוד. שיקולי פשטות העיצוב אינם מהווים שיקול מרכזי. לעיתים, מערכות תפעוליות מורכבות מבוססות על מאות רבות של טבלאות אותן יש להכיר היטב, כדי לעדכן בצורה נכונה את בסיס הנתונים. מפתחי התוכנה עוברים תקופות הכשרה מתאימות עד ללימוד מבנה בסיס הנתונים, וגם אז הם בדרך כלל מתעניינים בקטעים מאוד מוגבלים שלו. זאת, מכיון שכל תנועה מטפלת, בדרך כלל, במספר קטן של טבלאות.

מסלולי גישה ידועים מראש (Predefined Access Path)

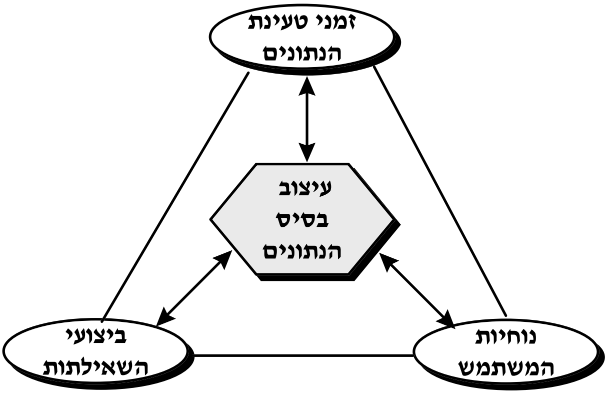
מכיון שמערכות תפעוליות מבצעות מספר גדול מאוד של תנועות בעלות אופי זהה (למשל, במערכות של חברות תעופה מתבצעת תנועה של רישום הזמנה לכרטיסי טיסה עשרות אלפי פעם ביום) ניתן לתכנן באופן מדויק מהו המסלול אותו תעבור התנועה בבסיס הנתונים (Data Base Navigation). ידיעה מראש זו מאפשרת למעצב בסיס הנתונים לקבל החלטות איזה אינדקסים לבנות, היכן לוותר על מבנים מנורמלים לטובת מבנים מנורמלים פחות, ולהשתמש במנגנונים נוספים שבסיס הנתונים מספק. לעומת מצב זה, מחסן הנתונים מתמודד עם שני סוגים של שאילתות - שאילתות ידועות מראש ושאילתות מזדמנות. עבור שאילתות ידועות מראש ניתן להכין מסלולי אופטימיזציה מתאימים. הבעיה הגדולה היא בשאילתות המזדמנות. כאן אין למעצב כל יכולת לצפות מראש כיצד השאילתה תתנהג.

התמקדות בנתונים שוטפים ולא היסטוריים

מערכות תפעוליות ממוקדות בעיקר בניהול היומיומי של הארגון. לכן, אך טבעי הוא הדבר שהנתונים הנדרשים הם, בדרך כלל, הנתונים השוטפים (מה רמת המלאי הנוכחית, מה מצבה של ההזמנה כרגע, האם עדיין נותרו מקומות לטיסה הקרובה לפריס וכדומה). בדרך כלל, מערכות תפעוליות ממוקדות פחות בנתונים היסטוריים, אם כי יהיה זה לא נכון לומר שהם אינם מטפלים בהיסטוריה כלל. לדוגמה, במערכת חיוב וגבייה של חברת טלפונים נשמרים נתוני החשבון לפרק זמן של מספר חודשים, כדי לאפשר ללקוח לבצע בירורים שונים. בשל העובדה שנתונים היסטוריים מעניינים מערכות תפעוליות באופן מוגבל, גם נפחי הנתונים המנוהלים במערכות אלו מוגבלים. לעומתם, מערכות מחסן הנתונים המשמשות לתהליכי קבלת החלטות חייבות לשמור את הנתונים לפרקי זמן ארוכים, בעיקר כדי לאפשר ניתוחי מגמה (Trend Analysis). צבירת הנתונים ההיסטוריים מביאה לגידול מתמיד בנפח הנתונים המנוהל במחסן הנתונים, ומהר מאוד אנו ניצבים בפני אתגר מורכב מאוד, ניהול בסיסי נתונים ענקיים. טכניקות העיצוב של בסיסי נתונים ענקיים (VLDB - Very Large Data Bases) הן שונות, ומבוססות על מנגנונים שונים (כמו, למשל, חלוקת הנתונים למחיצות שונות וכדומה).

שיקולי עיצוב עיקריים בסביבת מחסן נתונים

לעומת שיקולי העיצוב שהוזכרו קודם לכן, בסביבת מחסן הנתונים הדגשים שונים לחלוטין. מכיון שבסביבת מחסן הנתונים לא מתבצע עדכון, אלא רק אחזור הנתונים, צריך מעצב בסיס הנתונים להתמקד במציאת האיזון בין שלושה שיקולים שונים, ולעיתים סותרים - נוחיות ופשטות המבנה מנקודת מבט המשתמש, זמני הביצוע של השאילתות וזמני טעינת הנתונים אל מחסן הנתונים.



תרשים 6.1: האילוצים בעיצוב בסיסי נתונים בסביבת מחסן נתונים.

נסקור בקצרה את שיקולי העיצוב העיקריים בסביבת מחסן נתונים.

נוחות המשתמש

מכיון שהמשתמשים בסביבת מחסן הנתונים פועלים באופן ישיר על בסיס הנתונים, יש חשיבות רבה לכך שהעיצוב שלו יהיה פשוט ומובן לכל. בסיסי נתונים מורכבים הדורשים רמה גבוהה של התמצאות בו הם, בדרך כלל, מתכון לכישלון בסביבת מחסן הנתונים. על המעצב לקחת עובדה זו בחשבון, ולעצב בסיס נתונים פשוט.

ביצועי שאילתות (Query Performance)

זמן ביצוע השאילתות מהווה גורם חשוב מאוד לעיצוב. למרות שהנטייה הראשונית היא להניח שבסביבת מחסן נתונים לא יהיה נורא כל כך אם שאילתה תרוץ עוד זמן מה, המציאות היא שלא בכל המקרים נכון הדבר. עבור אותם משתמשים המפעילים שאילתות קבועות בתדירות גבוהה, זמן תגובה קצר הוא בהחלט שיקול חשוב, כזה שיבטיח שהמשתמש אכן ינצל את מחסן הנתונים במלואו. עבור אוכלוסיית מנתחי המידע המפעילים שאילתות מורכבות ומזדמנות, זמן התגובה יכול לנוע ממספר דקות ועד מספר שעות. הבעיה היא, שבקלות רבה מאוד ניתן למצוא שאילתות שרצות עשרות שעות ועדיין אינן מסיימות את פעולתן. גם כאן יש להבין את מהות שאילתות אלה, ולנסות להפחית את זמן התגובה שלהן, תוך שימוש בטכניקות עיצוב שונות.

נחזור ונדגיש שאחת הבעיות העיקריות של מחסן הנתונים, המשפיעה באופן דרמטי על זמן התגובה, היא נפח הנתונים שעל השאילתה לסרוק עד למציאת התשובה.

זמן טעינת הנתונים (Data Load Time)

להבדיל ממערכות תפעוליות, המעדכנות את הנתונים באמצעות תנועות, מערכות מחסן נתונים מתעדכנות בדרך כלל ב**מנות**, או **אצוות** (Batch). מנות אלו נוצרות בתהליכי גזירת והכנת הנתונים, ובסיום תהליכים אלה מתבצעת טעינת כמויות גדולות של נתונים בבת אחת. לדוגמה, בחברת טלפונים לא גדולה סביר להניח שמדי יום תידרש טעינה של עשרות מיליוני רשומות של פירוט שיחות. מכיון שחלון הזמן לעדכון הנתונים הוא מוגבל, יש להשקיע מאמץ עיצוב מיוחד כדי לקחת בחשבון את כמות הרשומות שיש לטעון, ולוודא שניתן יהיה לסיים את הטעינה במסגרת חלון הזמן.

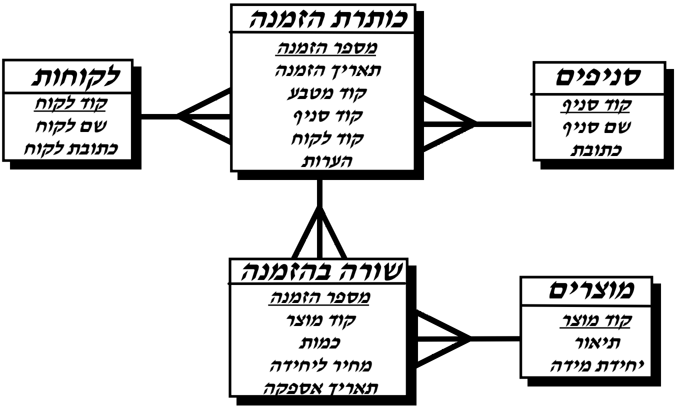
נירמול נתונים (Data Normalization) אינו יתרון בסביבת מחסן נתונים

נירמול נתונים (Data Normalization) היא טכניקת עיצוב ותיקה, שפותחה בתחילת שנות ה-70 על ידי Dr. E.F. Codd, אבי מודל הנתונים הטבלאי (Relational Data Model). מטרתה העיקרית של טכניקה זו היא להביא לפישוט מבנה הנתונים, על ידי הפחתה דרמטית בכפילות הנתונים. טכניקה זו מבוססת על פירוק יישויות ליישויות קטנות יותר, למניעת כפילות נתונים. תהליך הנירמול מגדיר את השיטה בה אוסף כלשהו של טבלאות מפורק לאוסף אחר בעל אותו תוכן מידע, כך שבכל טבלה מתקיים קשר הדוק בין מפתח הטבלה ליתר עמודות הטבלה. אוסף חדש זה של טבלאות הוא בעל אותו תוכן סמנטי כמו האוסף המקורי, אולם ללא כפילות נתונים. הפחתת כפילות הנתונים מביאה לכך שכל פיסת מידע מנוהלת בבסיס הנתונים פעם אחת בלבד. ניהול כל פיסת מידע פעם אחת בלבד מונעת את הצורך לעדכן מספר טבלאות כאשר הנתון משתנה. התוצאה המתקבלת במודל מנורמל היא, בדרך כלל, מספר רב מאוד של טבלאות פשוטות, בהן יודעת מערכת RDBMS לטפל באופן יעיל מאוד. המודל המנורמל המתקבל הוא מודל גמיש מאוד, אולם הוא מורכב יותר להבנה, בשל ריבוי הטבלאות.

בעיית הביצועים

הטבלאות עצמן קשורות ביניהן דרך עמודות בעלות משמעות סמנטית זהה, המופיעות בטבלאות שונות. החיבור חזרה ליישויות לוגיות מתבצע על ידי שימוש באופרטור הצירוף (Join), שהוא אופרטור יקר מבחינת ביצועים. בחלק מהמקרים, מוותרים מעצבי בסיס הנתונים על אלגנטיות המודל המנורמל, לטובת מודל המכיל מידה מסוימת של כפילות נתונים. הם עושים זאת כדי להשיג ביצועים טובים יותר. המודל הלא מנורמל דורש פחות פעולות צירוף, ולכן הוא בעל ביצועים טובים יותר באחזור נתונים. כל עוד ההתמקדות העיקרית היא בתהליך העדכון, כמובן שטכניקת הנירמול היא טכניקה טובה מאוד.

הבעיה העיקרית במודל המנורמל של בסיס הנתונים מתחילה ברגע שמבקשים לשאול שאילתות. כלומר, מתמקדים באחזור הנתונים, ופחות בעדכון שלהם. המידע על יישות לוגית אחת יכול להימצא במספר רב של טבלאות שונות. לדוגמה, אם נסתכל על טבלת הזמנות מלקוחות נמצא בה רק את קוד הלקוח, אבל לא את שמו וכתובתו. אלה נמצאים בטבלה נפרדת, טבלת הלקוחות. כמו כן, נמצא בטבלת הזמנות הלקוחות רק את קוד החנות בה בוצעה ההזמנה, אבל לא את שם החנות. שם החנות נמצא בטבלה נפרדת, טבלת החנויות. שיטה זו נוחה מאוד עבור טרנזקציות שמטרתן לעדכן את בסיס הנתונים. לדוגמה, אם כתובת הלקוח משתנה, כל שעלינו לעשות הוא לעדכן שורה אחת בטבלת הלקוחות, וזאת למרות שייתכן וללקוח זה יש מאות הזמנות פתוחות.



תרשים 6.2: מודל יישויות קשרים מנורמל.

תרשים 6.2 מציג את מודל היישויות-קשרים של בסיס נתונים העוסק ברישום הזמנות למוצרים. המודל משתמש במלבן לייצוג יישות (כמו סניף, מוצר וכד׳). בכל מלבן מופיע שם היישות ואחריו מוצג אוסף התכונות שלה. המפתח העיקרי של היישות מסומן בקו תחתי. בין היישויות מתקיימים קשרים, שבדרך כלל הם מסוג אחד לרבים (או 1:N, כפי שמקובל לכנותם). לדוגמה, בין היישות ״לקוח״ לבין היישות ״כותרת הזמנה״ מתקיים קשר כזה. כלומר, ללקוח אחד יכולות להיות קשורות הזמנות רבות, אולם כל הזמנה קשורה ללקוח אחד בלבד.

אם נבקש לקבל את כל הנתונים על הזמנת לקוח זה, ומכיון שהמידע על היישות הלוגית ״הזמנת לקוח״ מפוצל בין מספר רב של טבלאות, עלינו לצרף את כל הטבלאות הללו. בשפת SQL תתקבל שאילתה כגון:

```
1.  SELECT    *
2.  FROM      ORDER_HEADER OR, ORDER_LINE OL, CUSTOMERS C,
3.             PRODUCTS P, BRANCHES B
4.  WHERE     OR.CUSTOMER_ID = C.CUSTOMER_ID AND
5.            OR.ORDER_NUMBER = OL.ORDER_NUMBER AND
6.            OL.PRODUCT_ID = P.PRODUCT_ID AND
7.            OH.BRANCH_ID = B.BRANCH_ID
```

מעבר למורכבות, הנובעת מהפיצול למספר רב של טבלאות, והצורך של המשתמש להכיר את כל הטבלאות האלו, כדי להשתמש בהן בצורה נכונה במשפט SQL, מתעוררת כאן גם בעיית ביצועים. ידוע שפעולות הצירוף יקרות מבחינת צריכת משאבי מחשב. הבעיה מחריפה במיוחד אם הטבלאות המשתתפות בשאילתה הן ענקיות.

מדיון קצר זה ניתן לראות שטכניקת נירמול הנתונים, השמה דגש חזק מאוד על מניעת כפילות נתונים (בעיקר כדי למנוע בעיות בעת עדכון הנתונים), אינה מתאימה במיוחד למערכות שהדגש בהן הוא אחזור נתונים.

כפי שנראה בהמשך, אחד הרעיונות בעיצוב בסיס הנתונים בסביבת מחסן הנתונים הוא לוותר על רמה גבוהה של נירמול, או במילים אחרות להסיר את הנירמול (Denormalization). אם נשתמש בדוגמה הקודמת, נוכל לקבל טבלה אחת בלבד אשר תכיל את כל פרטי הזמנות הלקוח.

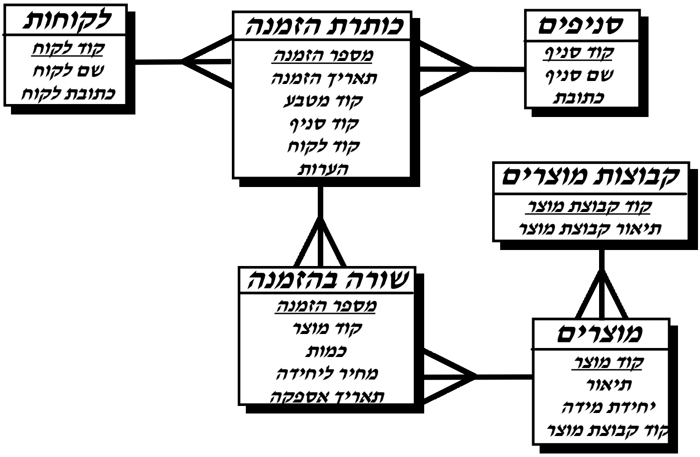


תרשים 6.3: מבנה נתונים לאחר הסרת הנירמול.

המחיר שנשלם במקרה זה הוא נפח האחסון, מכיון שחלק מהנתונים (כמו כתובת הלקוח) יחזרו על עצמם בכל אחת מהזמנות הלקוח. מצד שני, קיבלנו טבלה שטוחה ופשוטה בה כל הנתונים הדרושים מהווים חלק ממנה. לכן, למטרות אחזור הטבלה השטוחה מאוד יעילה, מכיון שאין צורך בביצוע פעולות צירוף.

בעיית ניהול מידע דינמי מול מידע היסטורי במחסן הנתונים

מכיון שמחסן הנתונים מנהל נתונים היסטוריים, לתהליך דה-נירמול הנתונים יש חשיבות נוספת. זו נובעת מההבדל בין מערכת תפעולית, הצריכה לייצג את הקשרים העדכניים בין הנתונים, ובין מחסן נתונים, שצריך לייצג את הקשרים כפי שהיו קיימים בעבר. כדי להסביר את הנושא נשתמש בדוגמה. נניח, שכל מוצר שייך לקבוצת מוצר מסוימת. בהתבסס על חוקי הנירמול, ננהל טבלה נפרדת של קבוצות מוצר ונוסיף מפתח זר לטבלת המוצרים, כדי לנהל את הקשר בין המוצר לבין קבוצת המוצר אליה הוא משתייך.



תרשים 6.4: מצב בסיס הנתונים המנורמל.

מעת לעת יכולה החברה לארגן מחדש את קווי המוצר שלה ולשנות את שיוכו של מוצר לקבוצת המוצר. סביר להניח שבמערכת התפעולית ננהל את השיוך העדכני ביותר. לעומת זאת, במחסן הנתונים המנהל את הנתונים ההיסטוריים, אנו רוצים לנהל את השיוך הנכון של כל מוצר לקבוצת המוצר שלו, על ציר הזמן. אם נבצע דה-נירמול של הטבלאות, ונכניס את קוד קבוצת המוצר אל הטבלה החדשה, נקבל את השיוך הנכון של המוצר כפי שהיה בעת ההזמנה.



תרשים 6.5: טבלה לא מנורמלת עם קוד ותיאור קבוצת מוצר.

בדומה, יכול תהליך הדה-נירמול לסייע לניהול נתונים המשתנים כל הזמן, כגון מחירי פריט וכדומה. הכנסת הנתון אל תוך הטבלה הלא מנורמלת, מקפיאה את הנתונים לאורך זמן. כפי שניתן לראות מדוגמאות אלו, תפיסת הנירמול מתאימה מאוד לניהול הקשרים הדינמיים בין הנתונים במערכות תפעוליות, אולם אינה מתאימה למחסן הנתונים המבקש להקפיא את מצבם של נתונים לאורך ציר הזמן.

סיכום ההבדלים העיקריים

ננסה לסכם, בתמציתיות, מדוע טכניקות העיצוב של בסיסי נתונים בסביבת עיבוד תנועות אינן מתאימות לעיצוב בסיס נתונים בסביבת מחסן נתונים.

- **אינו מתאים לניתוח מידע רב-מימדי** - כפי שהוסבר בפרק הקודם, הניתוח הרב-מימדי מהווה גורם חשוב ביכולת מנתחי המידע להפיק את המירב ממחסן הנתונים. עיצוב הנתונים המנורמל אינו תומך בצורת ההסתכלות הרב-מימדית של בסיס הנתונים, ולכן אינו מתאים לתהליכי תמיכה בקבלת החלטות.

- **קשה להבנה על ידי המשתמש** - כתוצאה משימוש בטכניקות של נירמול נתונים, העיצוב המתקבל יכול להיות מורכב מאוד ולהכיל עשרות, ולפעמים אף מאות טבלאות. משתמש רגיל אינו יכול להבין את מבנה בסיס הנתונים, ומכיון שבסביבת מחסן הנתונים המשתמש ניגש באופן ישיר לבסיס הנתונים (ללא ההגנה של יישומים כתובים מראש) המבנה המורכב המתקבל מציב קושי גדול מדי בפני המשתמשים, מה שבסופו של דבר עלול לגרום לכישלון מחסן הנתונים.

- **בעיות ביצועים** - בהיות העיצוב מבוסס על פירוק בסיס הנתונים לעשרות טבלאות, למניעת כפילות נתונים, נוצרת בעיית ביצועים חמורה בזמן שאילתות הצריכות לבצע מספר גדול מאוד של פעולות צירוף.

- **עקביות מקומית לעומת עקביות גלובלית** - עיצוב בסיס הנתונים התפעולי מבוסס על שמירה מתמדת של עקביות מקומית של התנועה (Local Transaction Consistency). כלומר, על המערכת לדאוג לכך שעם סיום התנועה, בסיס הנתונים יהיה עקבי ותקין. מכיון שמערכת תפעולית משרתת מאות ואלפי משתמשים, המבצעים תנועות בו-זמנית, סוגיית השמירה על העקביות של התנועה היא סוגיה מורכבת מאוד, ובסיס הנתונים מכיל מספר מנגנונים מיוחדים למטרה זו - קובץ יומן אירועים (Log File), יכולת גלגול אחורה אוטומטית של תנועה (Automatic Transaction Rollback), מנגנוני נעילת רשומות (Record Locking) ואיתור נעילות ללא מוצא (Deadlock Detection), וכדומה. לעומת זאת, בסיס הנתונים של מחסן הנתונים מבוסס על עקביות גלובלית (Global Consistency), מכיון שהנתונים הם אינטגרטיביים ברמת הארגון, ולא ברמת תהליך עדכון זה או אחר. מכיון שבסיס הנתונים מתעדכן רק במנות, ולא על ידי תנועות, רוב המנגנונים לשמירת עקביות התנועה הם חסרי חשיבות בסביבת מחסן הנתונים.

- **מימד הזמן חסר** - מנקודות מבטו של משתמש טיפוסי במחסן הנתונים, מימד הזמן הוא אחד המימדים העיקריים. כדי להבין ולנתח את המגמות (מה קרה החודש לעומת החודש הקודם, מה קרה החודש לעומת החודש המקביל לפני שנה, מגמות בין רבעונים וכדומה) מסתכלים מנתחי מידע ומקבלי החלטות על המידע בצורה של "פרוסות על ציר הזמן (Slice of Time View)". עיצוב הנתונים עבור מערכות תפעוליות מתבסס בעיקר על נקודת המבט של התנועות (Transactional View) שעל המערכת לבצע, ולכן הוא מתעלם כמעט לחלוטין ממימד הזמן.

- **אין כל התייחסות לנושא הסיכומים** - בסביבת מחסן הנתונים נושא הסיכומים (Aggregates) הוא בעל חשיבות רבה מאוד. טכניקות העיצוב של בסיסי נתונים בסביבות עיבוד תנועות אינן מתייחסות לסוגיה זו כלל, ואינן מספקות הנחיות כלשהן כיצד לנהל את הסיכומים.

- **התמקדות במצב העדכני ולא בהיסטוריה** - מערכת תפעולית ממוקדת מאוד במצב **השוטף** (Current) ובמצב **קרוב לשוטף** (Near Current), ולא בהיסטוריה. הצורך לנהל נתונים היסטוריים מציב אתגרים מיוחדים, גם מעבר לצורך לנהל נפחי נתונים גדולים. למשל, אם מתרחש שינוי במבנה הארגוני של ארגון כלשהו וסניף עובר ממחוז צפון למחוז מרכז, מבוצעים במערכות התפעוליות שינויים מסוימים כדי לשקף את המצב החדש. לעומת זאת, במחסן נתונים לא בהכרח נרצה לשבש את כל סטטיסטיקות העבר על ידי שיוך הסניף למחוז המרכז. אם נרצה להשוות ביצועים של המחוזות לאורך השנים, נצטרך למצוא דרכים כיצד לבטא את העובדה שעד תאריך מסוים השתייך הסניף למחוז הצפון, והחל מתאריך זה הוא שייך למרכז.

לסיכום ניתן לומר שמערכת, שעיקר יעודה הוא תמיכה בתהליכי קבלת החלטות וניתוח מידע, צריכה להיות מבוססת על בסיסי נתונים שנבנו ועוצבו מראש למטרות אלו.

מושגי היסוד - עובדות ומימדים

עיצוב רב-מימדי של בסיסי נתונים טבלאיים היא טכניקת עיצוב שמטרתה לבנות בסיס נתונים, במבנה ייעודי לנושא תמיכה בתהליכי קבלת החלטות, שצריכים לתמוך בעיקר באחזור נתונים וניתוח מידע. מבנה בסיס הנתונים המתקבל מתאים במיוחד לסביבות מחסן נתונים, הוא פשוט וקל להבנה על ידי המשתמשים והוא גם בעל ביצועים טובים, בכל הקשור לאחזור נתונים. טכניקה זו מבוססת על רעיונות הניתוח הרב-מימדי, המהווים שיטה פופולרית ונוחה לניתוח נתונים בתהליכי קבלת החלטות. היא משתמשת בבסיס הנתונים הטבלאי כתשתית למימוש הרעיונות.

בעוד שעיצוב בסיסי נתונים לסביבות תפעוליות מבוסס על מושגים כגון יישויות, קשרים, פירוק פונקציונלי ונירמול נתונים, הרי שהעיצוב הרב-מימדי משתמש באוסף שונה לחלוטין של מושגים - עובדות, מימדים, סיכומים וכדומה. היתרונות העיקריים של טכניקת עיצוב זו הם שהמודל המתקבל הוא פשוט וקל להבנה על ידי משתמשים, ובמובן מסוים טבעי עבורם, מכיון שהוא נבנה סביב רעיונות ברורים ומובנים כגון מוצרים,

לקוחות, מכירות, סניפים, זמן וכדומה. יתרון בולט נוסף בא לידי ביטוי בעובדה שהמודל יציב יותר לאורך זמן, ועמיד יותר לשינויים.

טכניקת עיצוב רב-מימדי של בסיסי נתונים באה להתמודד עם הניסיון לבנות מודל נתונים שיבטיח פשטות, יחד עם רמה גבוהה של ביצועים, למיגוון רחב של שאילתות בלתי ידועות מראש. יחד עם זאת, מה שמאפיין שאילתות אלה הוא שהן מבוססות על עובדות - כמה מוצרים רכש לקוח מסוים, מי הם היעדים המועדפים לשיחות טלפון בשעות הלילה בימי ג' בשבוע, מהם סוגי התביעות הנפוצות ביותר השנה בהשוואה לאלו של שנה קודמת, באיזו שנה היתה העונה המוצלחת ביותר מבחינת מכירות מנויים לתיאטרון וכן הלאה. על בסיס תכונה מעניינת זו של השאילתות הטיפוסיות בסביבת מחסן הנתונים, פותחה טכניקת עיצוב מיוחדת, העיצוב הרב-מימדי. טכניקה זו מחלקת את אוסף כל היישויות המנוהלות בבסיס הנתונים לשתי קטגוריות - עובדות ומימדים.

עובדות (Facts)

ברמה הבסיסית ביותר, מחסן הנתונים מנהל בעצם אוסף גדול של עובדות המייצגות את האירועים הבסיסיים הנובעים מהפעילות העסקית השוטפת של הארגון, והם בעלי משמעות מנקודת מבטו. בדרך כלל, אירועים אלה נרשמים במערכות התפעוליות השונות של הארגון, התומכות בתהליכים התפעוליים השונים.

עובדה (Fact) היא אירוע בעל משמעות עסקית לארגון והנובע מהפעילות השוטפת שלו. לכל עובדה יש מספר תכונות (Attributes) רלוונטיות.

להלן מספר דוגמאות של עובדות הנובעות מהפעילות העסקית השוטפת של הארגון:

- **מכירה** - אירוע של מכירת מוצר ללקוח ברשת מרכולים.
- **שיחת טלפון** - אירוע של התקשרות ברשת הטלפונים בחברת טלקומוניקציה.
- **תביעה** - אירוע של הגשת תביעה על ידי לקוח בחברת ביטוח.
- **תנועה בחשבון עו"ש** - אירוע של פעולה כלשהי בחשבון לקוח בבנק.
- **מנוי בתיאטרון** - אירוע של רכישת מנוי להצגות בתיאטרון.
- **שחרור חולה** - אירוע של שחרור חולה מבית חולים לאחר אשפוז.
- **קריאת שירות** - אירוע של קבלת תלונה מלקוח אודות תקלה במכשיר שרכש.
- **רכישת כרטיס טיסה** - אירוע של רכישת כרטיס טיסה ממקום ליעד כלשהו.
- **תקלה בקו ייצור** - אירוע של דיווח על תקלה בקו ייצור של מוצר כלשהו.

כפי שניתן לראות מדוגמאות אלו, אלה האירועים הבסיסיים ביותר בארגון והם מתרחשים מאות ואלפי פעמים ביום. המערכות התפעוליות רושמות אירועים אלה כחלק מתהליכי העבודה של הארגון.

לכל אחד מעובדות אלו יש מספר **תכונות** (Attributes) המאפיינות את האירוע. להלן מספר דוגמאות לתכונות בעלות משמעות עבור העובדות המוזכרות מעלה.

- **מכירה** - כמות הרכישה, מחיר הרכישה.
- **שיחת טלפון** - משך השיחה בדקות, מחיר השיחה, זמן תחילת שיחה.
- **תביעה** - סכום התביעה.
- **תנועה בחשבון עו״ש** - סכום התנועה.
- **מנוי בתיאטרון** - מספר כרטיסי המנוי שנרכשו, מספר ההצגות במנוי.
- **שחרור חולה** - משך האשפוז, הסכום לתשלום.
- **קריאת שירות** - הסכום ששולם לתיקון התקלה, משך הזמן שהמכשיר היה מושבת, משך התיקון.
- **רכישת כרטיס טיסה** - הסכום ששולם.
- **תקלה בקו ייצור** - משך התקלה.

כפי שניתן לראות, רוב התכונות הן נומריות, אם כי זה לא חייב תמיד להיות כך. כמובן שעובדות אלו אינן מעניינות במיוחד, מכיון שחסר לנו הֶקְשֵׁר העסקי שלהן. הֶקְשֵׁר (Context) זה יבוא ברגע שנוסיף את המימדים העסקיים.

את התכונות הנומריות ניתן לחלק לשלוש קטגוריות:

- **תכונות ברות חיבור** (Additive Attributes) - אלה תכונות נומריות שניתן לבצע עליהן פעולות אריתמטיות במיגוון רחב של חתכים שונים, ולקבל תוצאה בעלת משמעות. לדוגמה, סכום המכירה בש״ח. כל הפעולות בהמשך הן בעלות משמעות ונכונות - סה״כ המכירות לפי סניף, סה״כ מכירות ללקוח, סה״כ מכירות לתקופה מסוימת, ממוצע מכירות בין שתי תקופות, הפרש סה״כ מכירות בין תקופות וכדומה. דוגמה נוספת לתכונה ברת חיבור היא כמות המכירה, או מספר הדקות שמנוי שוחח בטלפון.
- **תכונות שהן ברות חיבור למחצה** (Semi Additive Attributes) - אלו תכונות נומריות שניתן לבצע עליהן פעולות אריתמטיות מסוימות בלבד. לדוגמה, היתרה בחשבון עו״ש בבנק. תכונה זו אינה ברת חיבור לאורך זמן כלומר, אין משמעות לחבר את היתרה בחודש אחד עם יתרה בחודש שני, מכיון שכל יתרה חדשה מחליפה את היתרה הקודמת. לעומת זאת, כן יש משמעות לחשב את היתרה הממוצעת לאורך זמן. דוגמה נוספת לתכונה ברת חיבור למחצה היא יתרה של פריט במלאי.
- **תכונות שאינן ברות חיבור כלל** (Non Additive Attributes) - אלו תכונות המנוהלות בטבלת העובדות, אולם אין כל משמעות לביצוע פעולות אריתמטיות עליהן. דוגמה לתכונה כזאת היא תאריך ביצוע תנועה בחשבון בנק.

מימדים עסקיים (Business Dimensions)

המימדים הם נושאים, או פרמטרים עסקיים הנותנים את המשמעות וההקשר לעובדות.

> **מימד** (Dimention) הוא נושא עסקי הנותן משמעות לעובדות. למימדים, בדומה לעובדות, יש מספר תכונות רלוונטיות.

אם נדמיין לרגע מרחב בו כל ציר הוא מימד עסקי כלשהו, ניתן להתייחס אל העובדות כאל ערכים המופיעים בהצטלבות בין הצירים. להלן מספר דוגמאות למימדים עסקיים רלוונטיים לכל אחת מהעובדות הנ"ל.

- ❖ **מכירה** - לקוח, מוצר, סניף, זמן, איש מכירות.
- ❖ **שיחת טלפון** - לקוח, זמן, סוג תעריף, סוג שיחה (רגילה, בינלאומית, גוביינא), טלפון מקור, טלפון יעד, מרכזת.
- ❖ **תביעה** - לקוח, זמן, סוג תביעה, סוכן ביטוח.
- ❖ **תנועה בחשבון עו"ש** - לקוח, זמן, סניף בנק.
- ❖ **מנוי בתיאטרון** - לקוח, זמן רכישת המנוי, זמן פג תוקף המנוי.
- ❖ **שחרור חולה מבית חולים** - חולה, זמן אשפוז, זמן שחרור, מחלקה, רופא אחראי.
- ❖ **קריאת שירות** - לקוח, מוצר מקולקל, זמן רישום הקריאה, זמן גמר תיקון, סניף מטפל.
- ❖ **רכישת כרטיס טיסה** - לקוח, זמן רכישה, זמן המראה, זמן נחיתה, שדה תעופה מקור, שדה תעופה יעד, סוג מחלקה.
- ❖ **תקלה בקו ייצור** - מוצר, קו הייצור, סוג תקלה, זמן דיווח תקלה, עובד מדווח.

בדומה לעובדות, לכל אחד מהמימדים יש מספר תכונות המאפיינות אותו. להלן מספר תכונות רלוונטיות למימדים השונים:

- ❖ **לקוח** - מספר לקוח, שם לקוח, כתובת, מספר לקוח, סוג לקוח, רמת אשראי.
- ❖ **זמן** - תאריך, יום בשבוע, יום בשנה, סוג היום (רגיל, חג, חול המועד).
- ❖ **מוצר** - קוד מוצר, תיאור מוצר, יחידת מידה.
- ❖ **שדה תעופה** - שם שדה התעופה, קוד שדה התעופה, אורך מסלול, רמת תחזוקה.
- ❖ **סניף** - שם סניף, כתובת הסניף, סוג הסניף, שם מנהל הסניף.
- ❖ **מחלקת אשפוז** - שם מחלקה, מספר מיטות, סוג המחלקה.

כפי שניתן לראות, תכונות המימדים הן מסוגים שונים - נומריים ואלפאנומריים. התכונות הלא נומריות הן בדרך כלל **תיאוריות** (Descriptive), מכיון שהן מתארות את המימד - שם לקוח, לקוח עסקי או פרטי, מוצר בעל אריזה סטנדרטית או לא וכדומה.

ייצוג יישויות על ידי טבלאות (Tables)

בבסיס נתונים טבלאי מיוצגות כל היישויות על ידי מבנה אחד ויחיד - הטבלה הדו-מימדית. לכל טבלה יש שם, מספר עמודות ומספר שורות. מבחינת בסיס הנתונים הטבלאי אין הבדל בין טבלה המייצגת עובדות לבין טבלה המייצגת מימד.

טבלת מוצרים

מספר מוצר	שם מוצר	סוג מוצר

טבלת מכירות

מספר מוצר	***מספר לקוח***	סכום מכירה	כמות מכירה

טבלת לקוחות

מספר לקוח	שם לקוח	כתובת	סוג לקוח

תרשים 6.6: עובדות ומימדים הן טבלאות.

לכל טבלה במודל הטבלאי יש **מפתח עיקרי** (Primary Key), המוגדר כעמודה אחת או יותר המזהה באופן חד משמעי שורה מסוימת בטבלה. לדוגמה, בטבלת הלקוחות המפתח העיקרי הוא מספר לקוח, המזהה באופן חד-ערכי מופע (שורה) של לקוח. בטבלת המכירות יהיה המפתח העיקרי **מפתח מורכב** (Multipart Key), המורכב ממספר המוצר וממספר הלקוח. זהו המצב הנפוץ ביותר, אולם יש מצבים בהם המפתח יכול להיות גם **מפתח מלאכותי** (נומרטור למשל), הניתן באופן אוטומטי על ידי המערכת התפעולית.

טבלת מוצרים

מספר מוצר	שם מוצר	סוג מוצר

טבלת מכירות

מספר מוצר	***מספר לקוח***	סכום מכירה	כמות מכירה

טבלת לקוחות

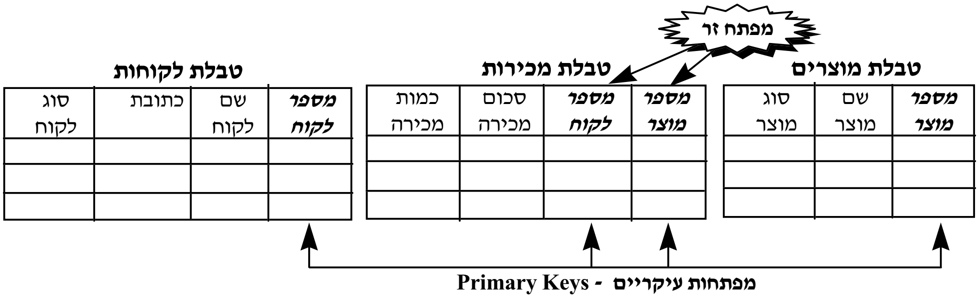
מספר לקוח	שם לקוח	כתובת	סוג לקוח

מפתחות עיקריים - Primary Keys

תרשים 6.7: מפתחות עיקריים.

בין טבלאות העובדות והמימדים מתקיימים קשרים לוגיים. לדוגמה, מכירה מסוימת שייכת ללקוח מסוים ובוצעה בסניף מסוים, שיחת הטלפון בוצעה על ידי לקוח מסוים בתאריך מסוים וכדומה. הדרך לייצג קשרים אלה בבסיס נתונים טבלאי היא על ידי שימוש ב**מפתחות זרים** (Foreign Keys). **מפתח זר** יוגדר כעמודה אחת, או יותר המופיעה בטבלה מסוימת, ומשמשת כמפתח עיקרי בטבלה אחרת.

בדוגמה שבתרשים 6.8, **מספר המוצר** בטבלה **מוצרים** הוא חלק מהמפתח העיקרי בטבלה **מכירות**, אך הוא גם מפתח זר. בדומה, **מספר לקוח** שהוא מפתח עיקרי בטבלת **לקוחות**, הוא החלק השני של המפתח העיקרי בטבלה **מכירות**, וגם הוא מוגדר בה כמפתח זר. מפתחות זרים אלה הם שיוצרים את הקשרים הלוגיים בין הטבלאות במודל הטבלאי.

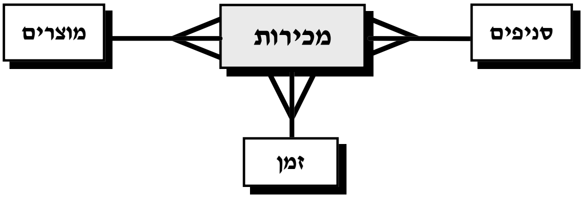


תרשים 6.8: קשרים בין טבלאות באמצעות מפתחות זרים.

סכמת כוכב (Star Schema)

מבנה

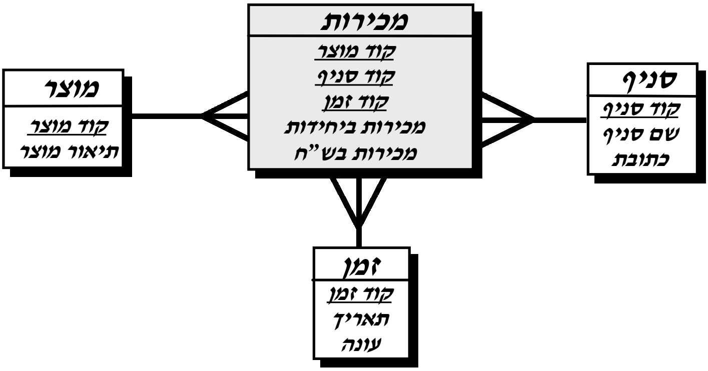
אם ניקח את טבלת העובדות ונשים אותה במרכז, וסביבה נשים את כל טבלאות המימדים נקבל מבנה מיוחד, של כוכב. במקרה זה מקובל לומר שבסיס הנתונים הטבלאי הוא בעל סכמת כוכב, להבדיל מסכמות רגילות הנובעות מעיצוב בסיס הנתונים עבור מערכות תפעוליות.



תרשים 6.9: מבנה סכמת כוכב.

המבנה המיוחד המתקבל קל מאוד להבנה על ידי משתמשים, בהשוואה למודל נתונים מנורמל. טבלת העובדות הנמצאת במרכז הכוכב, היא בדרך כלל הטבלה הגדולה ביותר, מכיון שהיא מכילה שורה עבור כל אירוע. גודל טבלת העובדות נקבע, בעיקר, על ידי כמות האירועים לתקופת זמן, ומספר תקופות הזמן שיש לנהל במחסן הנתונים. יחסית לטבלת העובדות, טבלאות המימדים קטנות יותר.

לכל אחת מטבלאות המימדים יש מפתח עיקרי (כמו קוד מוצר בטבלת המוצרים וקוד סניף בטבלת הסניפים). כל העמודות האלו מופיעות גם בטבלת העובדות, ומרכיבות את המפתח העיקרי (Multipart Primary Key). בנוסף, כל אחת מהעמודות המרכיבות את המפתח העיקרי היא גם מפתח זר, כלפי טבלאות המימדים. מבנה זה משקף קשרי 1:N, בין טבלאות המימדים לטבלת העובדות, או לחילופין קשרי N:M, בין טבלאות המימדים לבין עצמן. לטבלת העובדות יכול להיות מפתח עיקרי משלה (כמו למשל מספר מכירה), או לחילופין מפתח עיקרי המורכב מכל המפתחות הזרים. התרשים הבא מציג את סכמת הכוכב, תוך פירוט המפתחות העיקריים והעמודות של כל אחת מהטבלאות.



תרשים 6.10: פירוט טבלאות, עמודות ומפתחות בסכמת כוכב.

אני מניח שבשלב זה נראית טבלת הזמן מוזרה במקצת, ואתם בוודאי שואלים את עצמכם "מדוע לא להוסיף עמודה המכילה את תאריך המכירה בטבלת המכירות, ולוותר על טבלת הזמן?". בשלב זה, נשאיר שאלה זו פתוחה. כאשר נחזור אליה בהמשך נראה שיש הגיון רב בהגדרת מבנה זה.

הגדרת הטבלאות בשפת SQL מתבצעת בצורה רגילה. בהמשך, מופיעה הגדרת טבלת המוצרים וטבלת המכירות. ספרור השורות מופיע רק לשם נוחיות, והוא אינו חלק מההגדרה.

```
1.  CREATE TABLE PRODUCTS
2.      (PRODUCT_ID  INTEGER,
3.       PRODUCT_NAME   VARCHAR (20),
4.      PRIMARY_KEY (PRODUCT_ID))
```

```
1.  CREATE TABLE SALES
2.      (PRODUCT_ID         INTEGER,
3.       BRANCH_ID          INTEGER,
4.      PERIOD_ID           INTEGER,
5.      SALE_QUANTITY  DEC (6,2),
6.      SALE_AMOUNT      DEC (10,2),
7.      PRIMARY kEY (PRODUCT_ID, BRANCH_ID, PERIOD_ID),
8.      FOREIGN KEY (PODUCT_ID) REFERENCES PRODUCTS ,
9.      FOREIGN KEY (BRNCH_ID) REFERENCES BRANCHES ,
10.     FOREIGN KEY (PERIOD_ID) REFERENCES PERIODS )
```

בהגדרת טבלת המוצרים מגדיר משפט 4 את המפתח העיקרי. בהגדרת טבלת המכירות, שהיא טבלת העובדות, מגדיר משפט 7 את המפתח העיקרי. מפתח זה מורכב משלוש עמודות. לאחר מכן, משפטים 8 עד 10 מגדירים את המפתחות הזרים ליצירת הקשר הלוגי בין טבלת העובדות לכל טבלאות המימדים שלה.

לכל אחת מהטבלאות ניתן להגדיר את האינדקסים שלה.

```
1.  CREATE INDEX PRODUCT_IDX ON PRODUCTS (PRODUCT_ID)
2.   CREATE INDEX SALES_IDX ON SALES (PRODUCT_ID, BRANCH_ID, TIME_ID)
```

שאילתות כוכב (Star Query)

השימוש בשפת SQL מאפשר לשאול שאילתות רבות ומגוונות, כאשר כולן מבוססות על אותם עקרונות - מגדירים אילוצים על עמודות בטבלאות המימדים, ומבצעים צירוף של כל טבלת מימד עם טבלת העובדות. נציג שאילתה המציגה עבור כל מוצר את סך כל המכירות בעונת האביב, אם המוצר הוא בקבוצת מוצרי היוקרה. את התוצאה יש למיין לפי סכום הרכישה, בסדר יורד.

```
1. SELECT      PRODUCT_NAME, SUM (Sales_Amount) AS TOT_SALES
2. FROM        SALES S, PRODUCTS P, PERIODS T
3. WHERE       P.PRODUCT_CATEGORY = 'Prestige' AND
4.             T.SEASON = 'Spring' AND
5.             P.PRODUCT_ID = S.PRODUCT_ID AND
6.             T.PERIOD_ID = S.PERIOD_ID
7. GROUP BY    PRODUCT_NAME
8. ORDER BY    TOTAL_SALES
```

כפי שניתן לראות, שורות 3 ו-7 מגדירות את האילוצים על טבלאות המימדים השונות, ושורות 5 ו-6 מבצעות את פעולת הצירוף של כל אחת משתי טבלאות המימדים אל טבלת העובדות. אחת השאלות המעניינות היא כיצד תבוצע האופטימיזציה של שאילתה זו. נשאיר נושא זה לשלב מאוחר יותר.

יתרונות עיקריים של סכמת הכוכב

לסכמת הכוכב יש מספר יתרונות חשובים בסביבת מחסן נתונים. אם היה נעשה שימוש בטכניקות עיצוב רגילות, המבוססות על טכניקות יישויות וקשרים (ER), יתרונות אלה לא היו באים לידי ביטוי.

- **מבנה קבוע** - המבנה הקבוע של סכמת כוכב, בו טבלת העובדות היא במרכז וסביבה אוסף של טבלאות מימדים, מאפשר בניית כלי קצה (כגון מחוללי שאילתות או מחוללי דוחות) היכולים להתבסס על מבנה זה. מחד, יספקו כלים אלה ממשק משתמש פשוט ומאידך יאפשרו לרכיב האופטימיזציה להגדיר מסלולי גישה מהירים ויעילים, המביאים בסופו של דבר, לביצועים טובים. לדוגמה, מכיון שרוב האילוצים בשאילתה הם סביב תכונות המופיעות בטבלאות המימדים, יכולים כלים אלה להציג בצורה נוחה את התכונות, ולאפשר למשתמש להגדיר את האילוצים (ערך מסוים, קבוצת ערכים, טווח וכדומה).
- **סימטריות** - כל אחד מהמימדים יכול לשמש נקודת כניסה של השאילתה. סימטריות זו מאפשרת היענות זהה למספר גדול ביותר של שאילתות שונות, וכל זאת ללא ידיעה מראש מהן השאילתות הנפוצות יותר או פחות במחסן הנתונים.
- **היענות נוחה לשינויים** - המודל המתקבל מסוגל להסתגל במהירות רבה, ובנוחיות לשינויים עתידיים. שינויים אלה יכולים להיות הוספת תכונות חדשות לטבלאות העובדות או המימדים, הוספת מימדים חדשים, ניהול היסטוריה של שינויים וכדומה. נושא זה מוצג ביתר הרחבה בהמשך.

מבנה המפתחות של הטבלאות (Key Structures)

הקשרים הלוגיים בין הטבלאות בבסיס נתונים טבלאי מבוססים על מפתחות עיקריים ומפתחות זרים. טבלת העובדות, הנמצאת בלב סכמת הכוכב, מכילה את כל המפתחות הזרים ליצירת הקשרים עם טבלאות המימדים. קביעת המפתחות של כל יישות בבסיס הנתונים היא קביעה חשובה, וכל שינוי במפתחות יכול לגרום לשינויים משמעותיים בבסיס הנתונים. בפני מעצב בסיס הנתונים עומדות שתי חלופות - שימוש במפתח התפעולי של היישות, או שימוש ב**מפתח חלופי**, **מלאכותי** (Surrogate Key) מיוחד, המשמש רק את מחסן הנתונים. הנטייה הטבעית היא להשתמש במפתחות הרגילים של כל מימד כמפתח זר, כלומר להשתמש במפתחות של יישויות אלו כפי נקבעו במערכות התפעוליות. לגישה זו שתי בעיות עיקריות:

- **נפח אחסון גדול** - לעיתים נקבעים מפתחות היישויות במערכות התפעוליות בשל אילוצים שונים. מכיון שקיימת נטייה (לא נכונה) להכניס משמעות למפתחות, מתקבלים לעיתים מפתחות ארוכים, ומאוד לא נוחים לשימוש. לדוגמה, קוד מוצר יכול בקלות להגיע ל-12 תווים, למרות שבמציאות אין יותר מ-10,000 מוצרים. אורך המפתח נקבע בשל הניסיון לתת משמעות לקוד המוצר - 3 תווים לקבוצת המוצר, 3 תווים למחלקה המייצרת, והיתר למספר מזהה. שימוש במפתחות ארוכים אלה מגדיל באופן משמעותי את נפח טבלת העובדות.

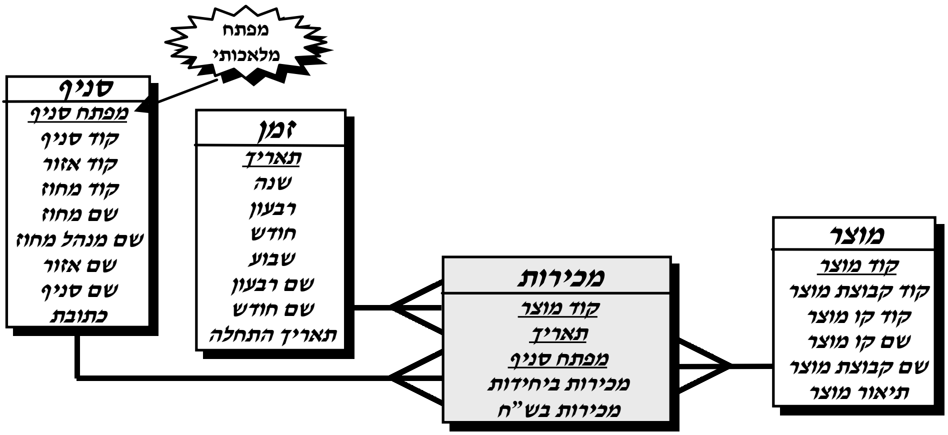
- **שינוי במפתח** - בשל הכנסת המשמעות למפתח, צריך לשנות את המפתח בעקבות שינויים במערכות התפעוליות או שינויים בתהליכי העבודה. לדוגמה, ייצורו של מוצר מסוים במחלקה אחת מופסק ומועבר למחלקה אחרת. דוגמה אחרת יכולה להיות הכנסת משמעות למספר סניף, כדי לשקף את השיוך הארגוני שלו: 2 תווים למחוז ו-2 תווים למספור רץ בתוכו. הבעיה יכולה להתעורר כתוצאה משינוי במבנה הארגוני, בו סניף עובר ממחוז למחוז, או מספר הסניפים מחוז עולה על 99.

- **גרירת מפתחות** - לעיתים, ליישות אין מפתח חד משמעי, אלא אם גוררים אליה מפתח של יישות אחרת (בדרך כלל תהיה זו יישות הנמצאת מעליה בקשר ההירככי). לדוגמה, ייתכן שבמערכת התפעולית נקבע שמספר המוצר יהיה חד משמעי רק בקבוצת המוצר אליה הוא שייך. משמעות הדבר, שמפתח המוצר הוא שילוב של קוד קבוצת המוצר ומספר המוצר. מעבר לכך שמפתח משולב זה גדול ולא נוח לשימוש, הסכנה הגדולה יותר היא שייתכנו שינויים בקשרים היררכיים אלה, במערכת התפעולית. העברה של מוצר מקבוצת מוצר אחת לשנייה מצריכה שינוי מפתח המוצר במחסן הנתונים, דבר מורכב מאוד לביצוע.

- **מפתחות תפעוליים שונים לאותה יישות** - לעיתים, נתקלים במצבים בהם אותה יישות מנוהלת במערכות תפעוליות שונות, ובכל אחת מהן עם מפתח שונה. לדוגמה, מספר לקוח במערכת העו״ש של בנק יכול להיות שונה ממספר הלקוח של אותו לקוח במערכת תוכניות החיסכון.

בשל חשיבות קביעת המפתחות, והקושי לבצע שינויים במפתחות, יש לשקול אפשרות נוספת והיא בניית מפתחות מלאכותיים, או כמו שהם נקראים לעיתים **מפתח מחסן נתונים** (Warehouse Keys). מפתחות אלה חסרי משמעות עבור היישויות במחסן הנתונים. השיקול העיקרי לשימוש ב**מפתח מלאכותי** הוא הערכת ההסתברות שיחולו שינויים במפתח הטבעי. לדוגמה, אם במערכת התפעולית מספר הסניף הוא בעל משמעות, מומלץ לשקול שלא להשתמש בו כמפתח, אלא לייצר מפתח מלאכותי עבור טבלת המימד של הסניף.

שימוש במפתחות מלאכותיים נותן רמה גבוהה יותר של גמישות ועמידות לשינויים, נושא שהוא בעל חשיבות רבה בסביבת מחסן הנתונים, שצריך לנהל עומק היסטורי רב. במובן מסוים, ניתן לומר שמפתח מלאכותי מוסיף רובד של הפשטה בין מחסן הנתונים לבין המערכות התפעוליות. רובד זה ימנע את הצורך בשינוי מפתחות בטבלאות מחסן הנתונים בכל פעם שמתבצע שינוי במערכת התפעולית. כמו כן, בשל היות המפתחות המלאכותיים קצרים יחסית, נפח האחסון ונפח האינדקסים יהיה קטן יותר, דבר שיביא לניצול טוב יותר של משאבי המחשב.



תרשים 6.11: מבנה כוכב עם טבלת סניפים המכילה מפתח מלאכותי.

בתרשים 6.11 המפתח המלאכותי היחידי הוא של הסניף. יתר המפתחות - קוד המוצר, מספר לקוח והתאריך הם מפתחות טבעיים של היישות.

ההחלטה על שימוש במפתחות מלאכותיים צריכה להיות סלקטיבית, מכיון שהיא מכניסה סיבוכיות נוספת בשלב הכנת הנתונים לטעינה למחסן הנתונים. בדוגמה הנ"ל, בעת טעינת סניפים למחסן הנתונים, יש לייצר מפתחות מלאכותיים. למשל, על ידי ניהול נומרטור מיוחד אותו יש לקדם עם הכנסת רשומה חדשה. הבעיה מורכבת עוד יותר בעת טעינת טבלת העובדות. המורכבות נובעת מכך שעל כל רשומת אירוע המגיע מהמערכת התפעולית יש להוסיף עיבוד, המבצע את התרגום מהמפתח התפעולי למפתח המלאכותי. לדוגמה, עבור כל רשומה המכילה את סך כל המכירות היומיות של כל מוצר בכל סניף ומתקבלת מהמערכת התפעולית, יש לבצע תרגום ממספר הסניף למפתח המלאכותי של הסניף, על ידי חיפוש בטבלת תרגום מיוחדת המנוהלת על ידי תהליך הגזירה והטעינה.

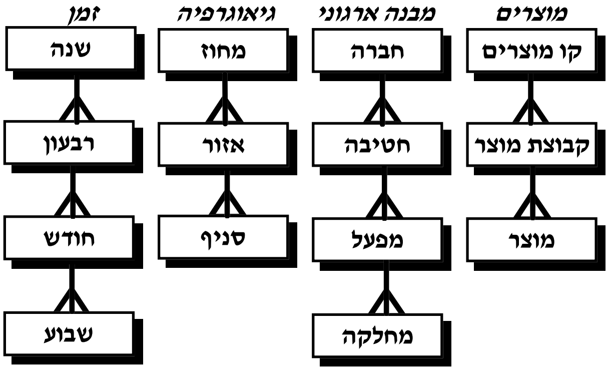
סוגיות מיוחדות בנושא טבלאות המימדים

הירקכיות של מימדים (Dimensions Hierarchy)

אם נתבונן במימדים העסקיים נראה שלרובם יש מבנה פנימי של הירקכיה, המתארת את הקבצות המימד ברמות שונות. נתבונן במימדים הבאים ונראה מספר הירקכיות אפשריות בכל אחד מהם :

- ❖ **זמן** - יום, שבוע, חודש, רבעון, שנה
- ❖ **מוצר** - מוצר, קבוצת מוצרים, כל המוצרים
- ❖ **סניף** - סניף, אזור, מחוז, כל החברה
- ❖ **מחלקה** - מחלקה, אגף, חטיבה, כל הארגון

התרשים הבא מציג את הירקכיות המימדים בצורה גרפית.

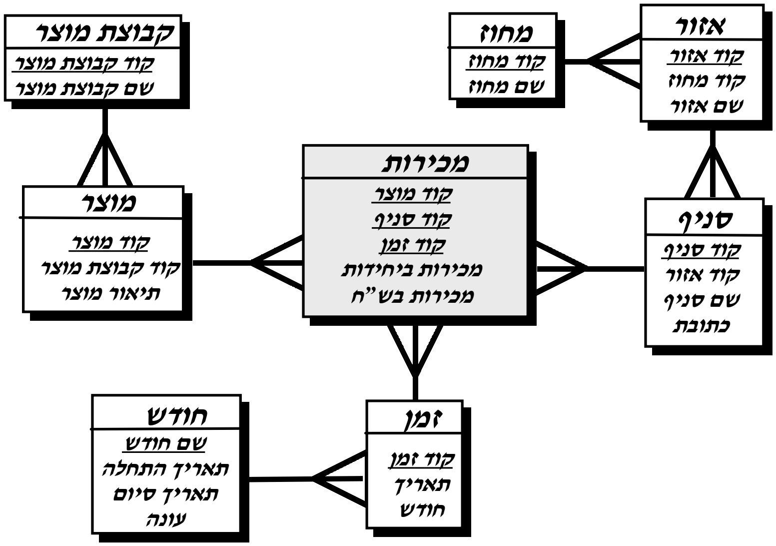


תרשים 6.12: הירקכיות מימדים.

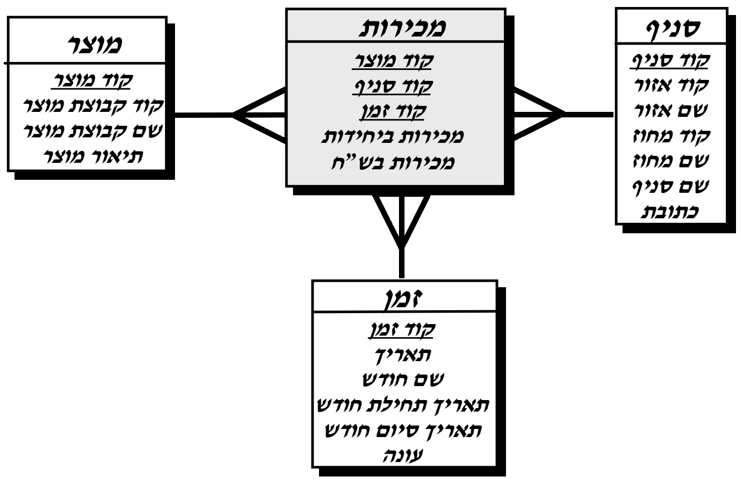
מבחינת טכניקות העיצוב, המבססות על נירמול נתונים, ברור שכל הירקכיה כזו תיוצג על ידי אוסף טבלאות, תוך יצירת קשרי 1:N ביניהן. ביצוע נירמול של הירקכיות המימדים יביא לקבלת מבנה כוכב מורכב יותר ממבנה הכוכב הרגיל.

המבנה המוצג בתרשים 6.13 מניח שהמפתחות בהירקכיה הם חד משמעיים - לדוגמה, קוד המוצר הוא חד משמעי באופן גלובלי, כלומר הוא אינו יכול לחזור על עצמו בקבוצות מוצר שונות. אם זה לא המצב אזי יש לגרור את המפתח מהרמה הגבוהה בהירקכיה אל הרמה הנמוכה יותר, וכמובן גם אל טבלת העובדות. בדוגמה שלנו משמעות הדבר שמפתח טבלת המוצרים יהיה מורכב מקוד המוצר וקוד קבוצת המוצר, וקוד קבוצת המוצר יוכנס גם לטבלת העובדות.

ברור שהמבנה המתקבל הוא מורכב יותר, ודורש מהמשתמש לכתוב שאילתות מורכבות יותר, המבוססות על מספר רב יחסית של פעולות צירוף. כדי למנוע הוספת רמת סיבוכיות נוספת זו לעיצוב מחסן הנתונים, מקובל להשתמש במבנה **לא מנורמל** (Denormalized).



תרשים 6.13: סכמת כוכב עם מימדים מנורמלים.



תרשים 6.14: מבנה כוכב לא מנורמל.

מבנה הכוכב הלא מנורמל המתקבל הוא פשוט, מנקדות מבטו של המשתמש, ויעיל יותר, מכיון שאין צורך במספר רב כל כך של פעולות צירוף. המחיר שנשלם, כדי לקבל מבנה פשוט זה, הוא הגדלת נפחי האחסון של טבלאות המימדים (בשל כפילות הנתונים שהכנסנו למודל). לדוגמה, שם המחוז יופיע בכל הסניפים השייכים לאותו מחוז. כמו כן, בניית מבנה נתונים לא מנורמל דורשת מאמץ נוסף, בעת בניית הנתונים שנגזרו מהמערכות התפעוליות, לקראת טעינתם למחסן הנתונים. מכיון שסביר להניח שהמבנה המנוהל במערכת התפעולית הוא מנורמל צריך, בעת הכנת הנתונים לטעינה למחסן הנתונים, לבצע עיבוד נוסף.

מימד הזמן (Time Dimension)

מדוע זהו מימד מיוחס

מימד הזמן הוא מימד מיוחד, הקיים בכל מחסן נתונים, ובמובן מסיים ניתן להתייחס אליו כאל **מימד יסוד** של מחסן הנתונים. מחסן הנתונים, מעצם הגדרתו, נועד לשמור היסטוריה, במטרה לאפשר ניתוחים אנליטיים של השינויים על ציר הזמן. המשתמש הטיפוסי במחסן הנתונים מסתכל על פרוסות זמן שונות ועורך השוואות ביניהן. מכיון שמימד זה מקבל חשיבות רבה, עיצוב נכון שלו יכול לפשט מאוד את השימוש במחסן הנתונים.

נכון להיום, רוב כלי האחזור אינם מזהים את מימד הזמן כמימד מיוחד, ולא מספקים תמיכה ישירה בביצוע שאילתות על ציר הזמן. סביר להניח שעם הזמן חלק גדול מהכלים לאחזור נתונים יטפל במימד זה באופן חכם כלומר, יזהו את מימד הזמן ויאפשרו ביצוע פעולות שונות המבוססות על סמנטיקה של המימד.

בעקרון קיימות טכניקות שונות לנהל את הזמן במודל הנתונים - ניהול של הזמן ישירות בטבלת העובדות, או הוספת טבלת מימד מיוחדת המנהלת את מימד הזמן בצורה מפורשת.

ניהול מימד הזמן ישירות בטבלת העובדות

זהו המצב הפשוט ביותר, ואולי הטבעי ביותר, עבור רוב המעצבים. הוספת עמודת **תאריך** לטבלת העובדות מוסיפה את מימד הזמן בצורה הטבעית והיעילה ביותר. מכיון שכל מערכות RDBMS תומכות באופן ישיר ב**טיפוס נתונים** (Data Type) מסוג **תאריך**, הוא מנוהל בצורה יעילה ונוחה בבסיס הנתונים. ניתן לבצע על תאריך זה את כל פעולות האריתמטיקה המותרות על תאריכים: למשל, החסרת תאריך מתאריך, הוספת מספר ימים לתאריך לקבלת תאריך חדש, וכדומה.

טכניקה זו טובה עבור מיקרים פשוטים, בהם שאילתות הפונות לטבלת העובדות מוסיפות אילוצים ישירים על תאריכים. למשל, כל המכירות שבוצעו בין תאריך מסוים לתאריך אחר כלשהו.

ניהול מפורש של מימד הזמן

החסרון העיקרי בניהול ישיר של תאריך בטבלת העובדות הוא, שהמשתמש הטיפוסי במחסן הנתונים משתמש במימד הזמן באופן מתוחכם מאוד, ופתרון פשוט זה אינו תומך בכך. לדוגמה, השוואת המכירות בין ימי ראשון שונים של השבוע השני בכל הרבעונים האחרונים, או בחינת המגמה בין המכירות בעונת קיץ השנה, לעומת עונת הקיץ בשנה קודמת, או בחינת אחוז הגידול במכירות ערב חג לעומת יום רגיל. טיפול ישיר בתאריכים עבור שאילתות אלו הוא מורכב מאוד, ולכן נדרש פתרון שונה.

הפתרון המקובל במחסני הנתונים הוא הוספה של טבלת מימד זמן מפורשת, ויצירת קשר בינה לבין טבלת העובדות. בטבלת מימד הזמן ניתן לנהל, עבור כל תאריך, מספר רב של תכונות שיסייעו מאוד בתהליכי התשאול השונים.

זמן
<u>תאריך</u>
יום בשבוע
שבוע בשנה
חודש
רבעון
שם רבעון
תקופה פיסקאלית
דגל יום חג
דגל יום ראשון בחודש
דגל יום אחרון בחודש
עונה
אירוע מיוחד

תרשים 6.15: מבנה טיפוסי של טבלת מימד הזמן.

בדוגמה המופיעה בתרשים 6.15 מופיעות מספר רב של תכונות רלוונטיות לתאריך: למשל, ציון האם זהו יום חג, ציון התקופה הפיסקאלית, ציון העונה, ציון האם זה יום תחילת חודש או סוף חודש, ציון אם זהו אירוע מיוחד (כמו למשל יום ה' בו יש משחק של מכבי תל אביב). תכונות אלו מאפשרות אחזור פשוט, על סמך נתונים אלה. טבלת מימד הזמן היא טבלה קטנה - 365 שורות לכל שנה. גם אם ננהל היסטוריה של מספר שנים, עדיין מדובר במעט שורות באופן יחסי. הדרך המקובלת היא, לפתוח מראש את כל השורות בטבלת מימד הזמן, למספר שנים קדימה.

השאילתה המופיעה בדוגמה הבאה מציגה את סך כל כמות המכירות של מוצר מסוים, בימי ב' ברבעון השני של שנת 1998.

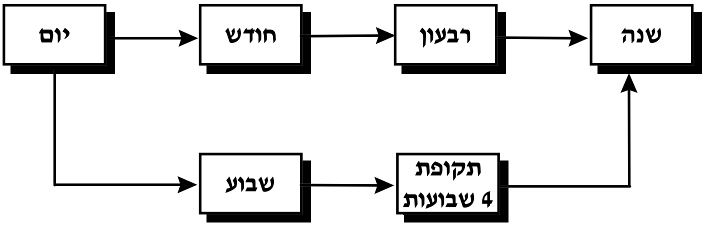
```
1.  SELECT      PRODUCT_NAME, SUM (Sales_Amount) AS TOT_SALES
2.  FROM        SALES S, PRODUCTS P, PERIODS T
3.  WHERE       T.QUARTER = 'Q2' AND
4.              T.YEAR = 1998 AND
5.              T.DAY_OF_WEEK = 'Monday' AND
6.              P.PRODUCT_ID = 'ABC-1345' AND
7.              P.PRODUCT_ID = S.PRODUCT_ID AND
8.              T.PERIOD_ID = S.PERIOD_ID
9.  GROUP BY  PRODUCT_NAME
10. ORDER BY  PRODUCT_NAME
```

ריבוי היררכיות במימד הזמן

למרות שממבט ראשון מימד זה נראה פשוט, הנושא יכול בהחלט להסתבך. למימד הזמן תכונה מעניינת, בכל הקשור להיררכיה. הבעיה העיקרית מתחילה מכך שהיררכיית הזמן אינה טרוויאלית, ולא ניתן לבצע סיכומים מלמטה כלפי מעלה (Summary Roll Up) באופן פשטני. כתוצאה מכך, גם השוואות בין תקופות הופכות למורכבות יותר. בדרך

כלל, מייצגת ההיררכיה מצב בו הרמות הנמוכות יותר מתקבצות בצורה ברורה לרמה גבוהה יותר, וחוזר חלילה. לדוגמה, הסניפים מתקבצים לאזור מכירות, אזורי המכירות מתקבצים למחוז ואילו המחוזות מתקבצים לכל החברה. דווקא במימד הזמן אנו מוצאים מצב חריג, בו ההקבצות אינן פועלות כפי שאנו מצפים. למשל, הימים מתקבצים לשבועות אולם שבועות אינם מתקבצים בצורה פשוטה לחודשים, מכיון שחודש יכול להתחיל באמצע שבוע. אם צריך לנהל את הנתונים גם לפי חודשים וגם לפי שבועות, יש ליצור שתי היררכיות מקבילות במימד הזמן. ברמה הבסיסית ביותר, שבוע עבודה מכיל 5 ימי עבודה. הבעיה היא שחודש יכול להסתיים גם באמצע שבוע. לכן עשויים להתקבל שבועות המכילים פחות מ-5 ימי עבודה. כתוצאה מכך גם יתקבלו חודשים המכילים 5 שבועות, ולא רק 4 שבועות, כמקובל.

פתרון אפשרי לבעיה זו היא ניהול היררכיות מקבילות במימד הזמן: היררכיה אחת בה היום מסתכם לחודש אליו הוא שייך, והיררכיה שנייה בה היום מסתכם לשבוע אליו הוא שייך.



תרשים 6.16: ריבוי היררכיות במימד הזמן.

מימדים מנוונים (Degenerate Dimensions)

לפעמים נוצר מצב בו קיים מימד שמאחוריו אין טבלת מימדים, אולם מפתח המימד מופיע בטבלת העובדות. לדוגמה, במודל נתונים העוסק במעקב אספקות נרצה לנהל את מספר ההזמנה בטבלת העובדות. עקרונית, ניתן להסתכל על ההזמנות כעל מימד האספקות, אולם אם אין לנו עניין לנהל מידע נוסף אודות ההזמנה, מימד זה הופך למימד מנוון.



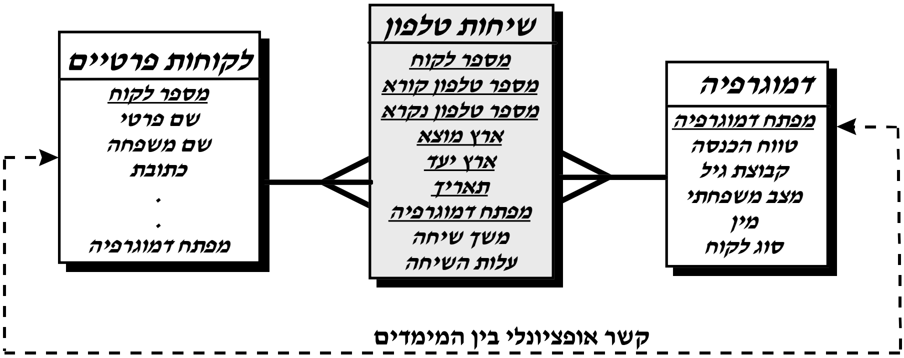
תרשים 6.17: מימד מנוון.

טיפול במימדים גדולים (Big Dimensions)

לפעמים, טבלת המימדים יכולה להיות טבלה ענקית כמו, למשל, במודל נתונים של חברת טלפונים המשרתת מספר מיליוני לקוחות. אחת האפשרויות היא להוסיף לכל לקוח נתונים דמוגרפיים כגון רמת הכנסה, גיל, מצב משפחתי וכדומה. כל תוספת של נתון אותו יש לנהל ברמת הלקוח הבודד, מייד מוכפלת במספר מיליונים, בשל הכמות הגדולה של לקוחות.

אפשרות אחרת היא לבנות טבלה דמוגרפית נפרדת, המכילה שורה אחת עבור טווחים בלבד - למשל רמת הכנסה בין 100,000 ל-150,000 ש״ח בשנה, קבוצת גיל בין 30 ל-40 ומצב משפחתי נשוי. ניתן ליצור מראש את כל הצירופים המעניינים את הארגון. כל שורה כזו מקבלת **מפתח מלאכותי** בטבלת הדמוגרפיה, שלעיתים ניתן לקרוא לה גם **מיני מימד דמוגרפי** (Demographic Mini Dimension). מפתח זה מוכנס כמפתח זר בטבלת השיחות, שהיא טבלת העובדות, וכן בטבלת מימד הלקוחות.

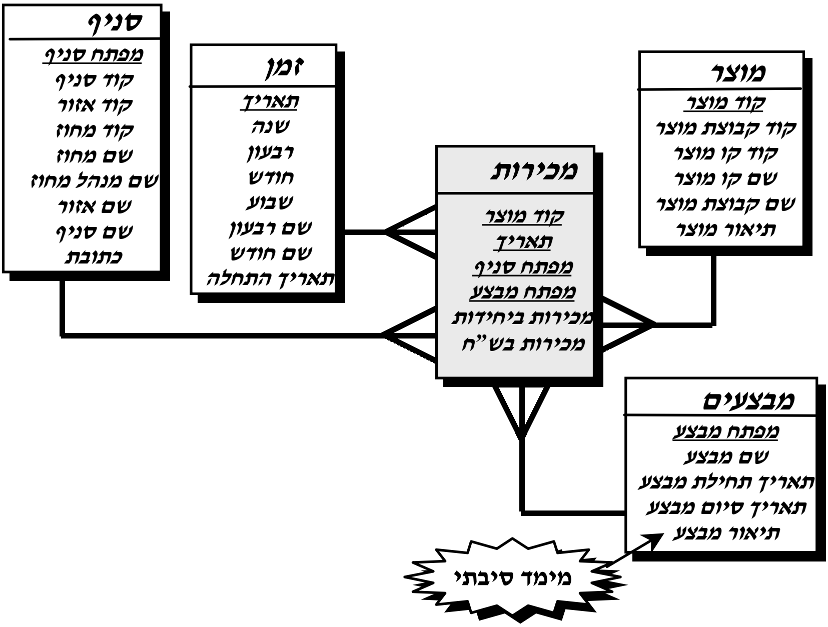
את השאילתות הדמוגרפיות ניתן לבנות על ידי פנייה ישירה מטבלת הדמוגרפיה לטבלת העובדות, ללא צורך בפנייה לטבלת מימד הלקוחות. למשל, שאילתה כגון סה״כ ההכנסות בגין שיחות של לקוחות השייכים לקבוצת גיל מסוימת, ובמצב משפחתי מסוים. עקרונית, ניתן לבנות גם מספר טבלאות **מיני מימד דמוגרפי**, כך שכל טבלה תכיל אוסף מסוים של תכונות דמוגרפיות רלוונטיות.



תרשים 6.18: פיצול טבלת מימדים לטבלת מיני מימד.

מימדים סיבתיים (Casual Dimensions)

מימד סיבתי הוא מימד המתאר מצב מסוים אשר יכול להוות סיבה להיווצרות אירוע שיירשם כעובדה בטבלת העובדות. לדוגמה, אירוע שיחת טלפון יכול להתרחש בעת שמבצע הנחות מסוים על שיחות בשעות הלילה היה בתוקף. דוגמה אחרת יכולה להיות שרכישת שירות חדש, כגון תא קולי, קרה בתקופה בה בוצע פרסום מסיבי אודותיו בעיתונות ובטלוויזיה. כמובן שאין כל דרך לדעת אם קיומו של מבצע זה או אחר היה הסיבה האמיתית לאירוע, אלא אם שואלים את הלקוח באופן ישיר. בכל מצב אחר אנו יכולים, במקרה הטוב, לחשוד שהמבצע הוא שגרם להתרחשות האירוע.



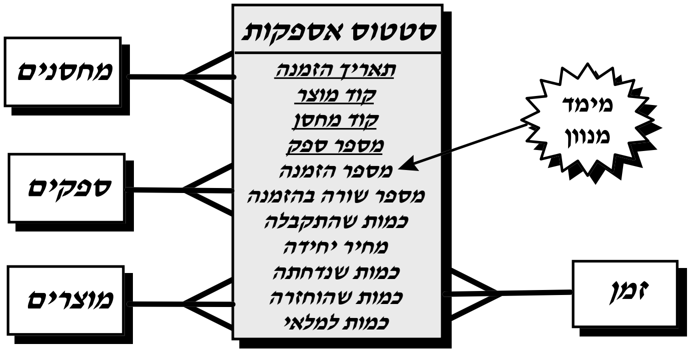
תרשים 6.19: מימד סיבתי.

קישור העובדות אל המימד הסיבתי יכול להתבצע במערכת התפעולית, או במחסן הנתונים. לדוגמה, המערכת התפעולית יכולה לרשום על כל תנועת מכירה גם את קוד המבצע. במצבים בהם המערכות התפעוליות אינן רושמות את הקשר למימד הסיבתי, צריך ליצור קשר זה, תוך כדי טעינת העובדות למחסן הנתונים. בתהליך הטעינה ייבדק אם בזמן האירוע היה מבצע כלשהו בתוקף, ואם כן - יבוצע הקישור. בתוך מימד המבצע נחזיק גם שורה אחת מיוחדת עבור מצב של "אין מבצע". כל העובדות יקושרו לשורה זו עבור מצבים בהם לא התקיים מבצע כלשהו.

קיומו של מימד סיבתי יכול לאפשר לארגון לשאול שאלות כגון - האם המבצע היה כדאי, האם היה שינוי במכירות כתוצאה מהמבצע.

ריבוי קשרים בין מימד לעובדות (Multiple Column Dimension)

לפעמים, מתקיים יותר מקשר לוגי אחד בין טבלת העובדות לבין טבלת מימד מסוימת. לדוגמה, עבור מודל נתונים העוסק במעקב אספקות, נרצה לנהל מספר תאריכים שונים הקשורים לאספקה מסוימת - תאריך ההזמנה, תאריך האספקה הראשונה ותאריך הקבלה האחרונה. במקרה זה, נגדיר את שלושת התאריכים האלה כמפתחות זרים כלפי טבלת מימד הזמן, ונקבל שלושה קשרים לוגיים בין שתי הטבלאות.



תרשים 6.20: ריבוי קשרים בין טבלת עובדות וטבלת מימד.

ריבוי קשרים זה מאפשר לנו לבצע שאילתות שונות כמו ניתוחי עונתיות באספקות, ניתוח קבלות ברבעון מסוים וכדומה.

סוגיות מיוחדות בנושא טבלת עובדות

הגדרת רמת הפירוט (Granularity)

אחד מנושאי העיצוב העיקריים של טבלת העובדות היא קביעת רמת הפירוט שלה. ככל שרמת הפירוט המנוהלת בטבלת העובדות גבוהה יותר, כך נפח הנתונים שינוהל במחסן הנתונים יהיה גדול יותר. במערכות תפעוליות רמת הפירוט המנוהלת היא הרמה הפרטנית ביותר הנדרשת - שיחה בודדת במערכת טלקומוניקציה, שורת הזמנה במערכת הזמנות, המכירה הבודדת במערכת מכירות וכדומה. בבואנו לעצב את טבלת העובדות של מחסן הנתונים, ההחלטה על רמת הפירוט אינה קשורה ישירות לרמת הפירוט המנוהלת במערכת התפעולית. היא קשורה לרמת הפירוט הנדרשת בתהליכי קבלת ההחלטות. מובן שלא ניתן לנהל רמת פירוט גבוהה יותר מאשר במערכת התפעולית, אולם בהחלט ניתן לנהל רמת פירוט נמוכה יותר. לדוגמה, במחסן נתונים של חברת טלקומוניקציה ניתן להחליט על רמות פירוט שונות. נציג שלוש חלופות שונות לרמת פירוט:

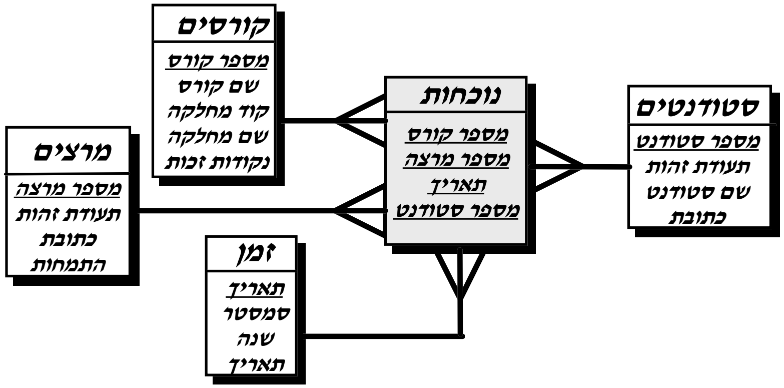
- **משך השיחה הבודדת** - לכל שיחה שהלקוח ביצע תישמר רשומה אחת.
- **משך השיחות של לקוח ביום** - לכל לקוח תישמר רשומה אחת ליום ובה יופיע מספר הדקות הכולל של משך השיחות באותו יום. במידת הצורך, ניתן לעדן רמת פירוט זו על ידי ניהול משך דקות השיחה לפי שלוש רצועות זמן ביממה (לדוגמה: שלוש רצועות זמן של 8 שעות כל אחת).
- **משך השיחות של לקוח בשבוע** - לכל לקוח תישמר רשומה אחת לשבוע ובה יופיע משך דקות השיחה הכולל במהלך השבוע. במידת הצורך ניתן לעדן רמת פירוט זו על ידי ניהול משך דקות השיחה בכל אחד מימי השבוע, באמצעות שבעה מונים נפרדים באותה רשומה.

כפי שניתן לשער, לכל אחת מרמות פירוט אלו יש השלכה שונה על גודל טבלת העובדות. נתבסס על מספר הנחות, כדי לחשב את גודל הטבלה. אם נניח שלקוח מבצע 10 שיחות ביום, נקבל בממוצע 300 שיחות לחודש, או 3,600 שיחות לשנה. אם לחברה יש 10 מיליון לקוחות ולכל שיחה יש לשמור 50 בתים, נקבל שעבור רמת הפירוט המקסימלית נצטרך לנהל 36 מיליארד שורות לשנה. גודל טבלה זו יהיה 180 מיליארד בתים, או 180 טרה-בתים (TeraByte).

זהו ללא ספק נפח נתונים גדול. ההחלטה לנהל רמת פירוט ליום בלבד, יכולה להקטין נפח זה באופן משמעותי. ככל שרמת הפירוט נמוכה יותר, נפח הנתונים יקטן, אולם כמוהו גם תקטן יכולת קבלת ההחלטות. למשל, אם רמת הפירוט היא משך השיחות ליום, לא ניתן לנתח מגמות ברמת השעה הבודדת.

טבלת עובדות נטולת עובדות (Factless Fact Table)

כל הדוגמאות שהוצגו עד לנקודה זו הכילו טבלת עובדות המורכבת משני מרכיבים עיקריים - אוסף של מפתחות זרים, היוצרים את הקשרים עם טבלאות המימדים, ואוסף של תכונות, המייצגות את הנתונים שיש לנהל לגבי כל אירוע. בנוסף למצבים אלה קיימים מצבים בהם טבלת העובדות אינה מכילה שום תכונה נוספת מעבר למפתחות הזרים של המימדים. במצב כזה טבלת העובדות משמשת רק ליצירת המפגש בין המימדים. הדוגמה הבאה מתייחסת למודל נתונים העוסק במעקב אחר נוכחות סטודנטים בהרצאות באוניברסיטה. המימדים המעניינים הם מרצים, סטודנטים, קורסים וזמן. שורה בטבלת העובדות מייצגת, במקרה זה, את האירוע המפגיש את ארבעת המימדים - כלומר סטודנט נוכח בהרצאה בקורס שניתן על ידי מרצה בזמן כלשהו. מעבר לעצם נוכחותו של הסטודנט בקורס, אין תכונות רלוונטיות נוספות.



תרשים 6.21: טבלת עובדות ללא עובדות.

מודל נתונים זה יכול לשמש לקבלת מידע אודות נושאים כגון - מהם הקורסים הפופולריים ביותר, כמה קורסים לקח כל סטודנט בכל אחת מהשנים בהם הוא לומד באוניברסיטה ועוד. נשים לב שרוב שאילתות אלו מבצעות תהליכי **מנייה** (Counting).

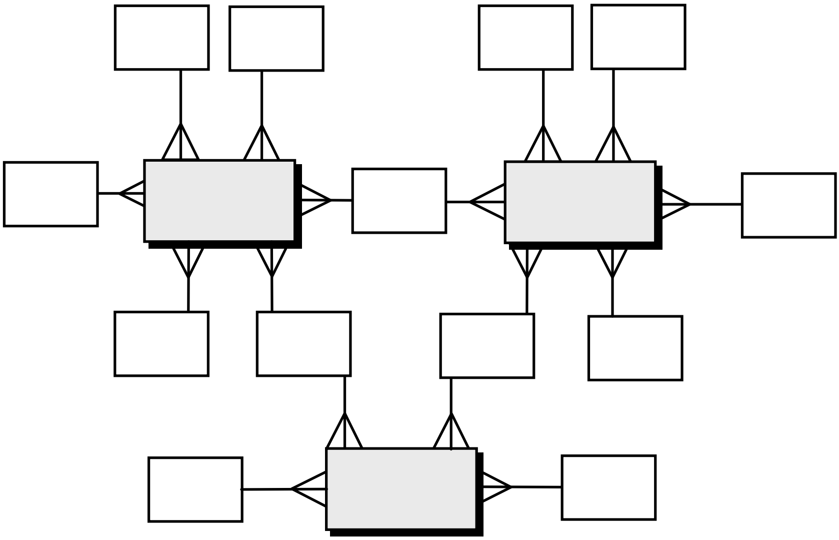
לדוגמה, השאילתה הבאה מציגה את מספר הסטודנטים שלמדו אצל כל מרצה בסמסטר מסוים.

```
1. SELECT      TEACHER_NAME, COUNT (PERIOD_ID)
2. FROM        TEACHERS T, ATTENDANCE A, PERIODS P
3. WHERE       SEMESTER = 'Fall 98' AND
4.             T.TEACHER_ID = A.TEACHER_ID AND
5.             P.PERIOD_ID = A.PERIOD_ID
6. GROUP BY  TEACHER_NAME
7. ORDER BY  TEACHER_NAME
```

מכיון שרוב השאילתות במודל זה הן בעלות אופי של מנייה, יש כאלה המוסיפים לטבלת העובדות עוד עמודה אחת, המכילה תמיד את הערך 1. כך, ניתן גם להשתמש בפונקציה Sum, לא רק ב-Count.

ריבוי טבלאות עובדות (Multiple Fact Tables)

עד כאן התייחסנו אל מחסן הנתונים כאל בסיס נתונים שיש בו כוכב אחד בלבד, כלומר יש בו טבלת עובדות אחת בלבד. המציאות מורכבת בהרבה, והסיכוי לבנות מחסן נתונים המכיל טבלת עובדות אחת בלבד, קלוש. במחסן נתונים אמיתי, בו יש מספר כוכבים וחלק מהמימדים משותפים למספר כוכבים, תתקבל למעשה סכמה מורכבת יותר. לעיתים, נקראת סכמה זו גם בשם **סכמת גלקסיה** (Galaxy Schema).

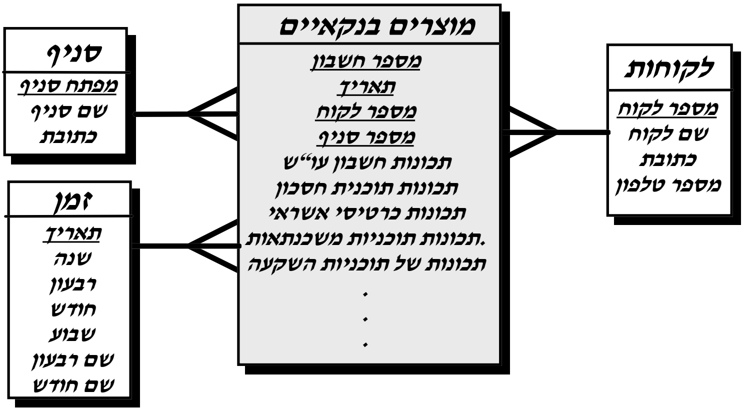


תרשים 6.22: סכמת ריבוי כוכבים.

נתבונן לדוגמה במחסן נתונים של בנק, המספק מיגוון רחב של מוצרים בנקאיים ללקוחותיו - חשבונות עו״ש, תוכניות חיסכון, תוכניות השקעה, כרטיסי אשראי, משכנתאות וכדומה. אחת השאלות הבסיסיות בבואנו לעצב את מודל הנתונים היא, האם נבנה טבלת עובדות אחת בלבד, או שלכל מוצר בנקאי נבנה טבלת עובדות נפרדת.

ממבט ראשון, ברור שלכל המוצרים הבנקאים המוזכרים יש מספר תכונות משותפות (כגון מי הלקוח, איזה סניף מנהל את המוצר, מה יתרת החשבון של הלקוח וכדומה). אולם, ברור גם שלכל מוצר יש תכונות האופייניות רק לו - לחשבון עו״ש יש תכונות כגון תקרת אשראי, ריבית, עמלות שירות מיוחדות, ואילו לחשבון חסכון יש תכונות כגון סכום הפקדה ראשוני, סכום הפקדה חודשי וכדומה. לכרטיסי אשראי יש תכונות כגון תאריך הנפקת הכרטיס, נקודות שנצברו במועדון לקוחות וכדומה. כלומר, לפנינו מספר מוצרים הטרוגניים.

חלופה אחת לעיצוב מודל הנתונים היא לבנות טבלת עובדות אחת, המכילה את האיחוד של כל התכונות מכל סוגי המוצרים הבנקאיים, כפי שמודגם בתרשים 6.23.

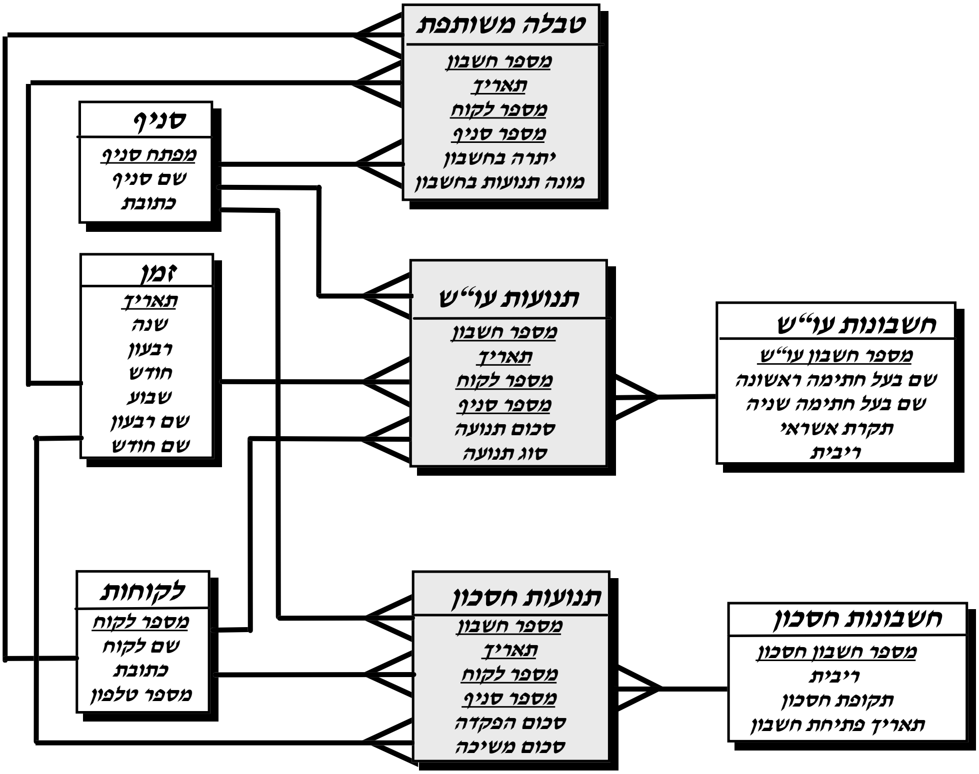


תרשים 6.23: טבלת עובדות אחת למוצרים הטרוגניים.

בניית טבלת עובדות אחת למצב בו קיימות מספר רב של עובדות הטרוגניות, מאפשרת בניית מבנה כוכב אחד. אבל זהו מצב בעייתי מאוד. טבלת העובדות תכיל מספר גדול מאוד של תכונות, כאשר ברוב השורות תהיינה חלק מהתכונות ריקות. מכיון שזו הטבלה הגדולה ביותר בבסיס הנתונים, ניתן לראות ששטחי האחסון אינם מנוצלים היטב. מעבר לכך, כל פעם שהבנק יחליט להוסיף מוצר בנקאי נוסף, נצטרך להרחיב את טבלת העובדות המרכזית, כדי לנהל את התכונות הייחודיות החדשות. עיצוב זה אינו עמיד לשינויים, ולכן אינו יעיל.

הפתרון הנכון למצב שתואר כאן הוא בנייה של מספר טבלאות נפרדות לכל אחד מהמוצרים השונים. בנוסף לטבלאות השונות המנוהלות עבור כל מוצר, ניתן גם לנהל טבלה אחת המשותפת לכל המוצרים - אם מצאנו שיש מספר תכונות שהן משותפות לכל המוצרים. ניהול טבלה משותפת כזו חוסך מקום אחסון, אולם מצד שני מסבך את השאילתות, מכיון שגם בשאילתות המתעניינות במוצר אחד בלבד נוצר הצורך לעבוד עם שתי טבלאות עובדות במקביל. מסיבה זו מומלץ לעיתים לנהל את הנתונים המשותפים בכל טבלה בנפרד, כך שהשאילתות תהיינה פשוטות יותר.

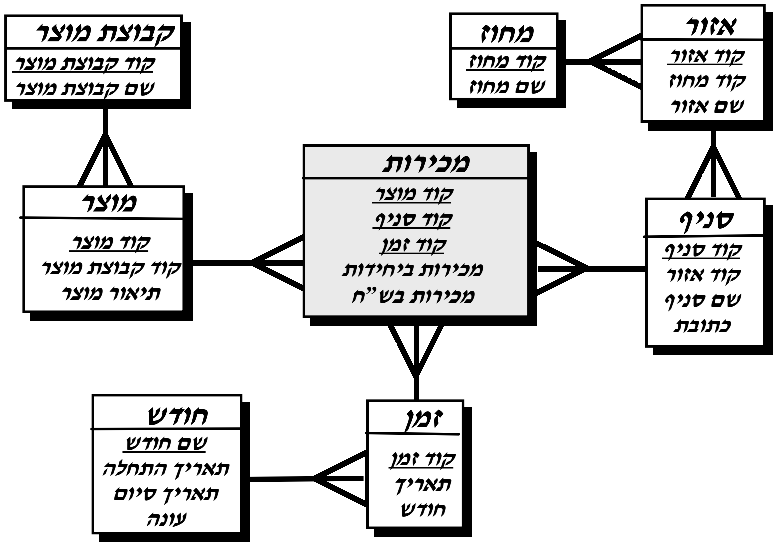
קיבלנו כאן מספר מבני כוכב שיש להם מספר מימדים משותפים - כמו לקוחות, סניפים, זמן - ומספר מימדים ייחודיים, כמו המוצר הבנקאי. זהו מבנה מורכב יותר מהמבנה הרגיל, אולם הוא מאפשר גמישות בהוספת מוצרים חדשים, ומנצל באופן יעיל יותר את שטחי האחסון. שאילתות המתייחסות למוצר אחד בלבד הן פשוטות. שאילתות המבקשות לקבל בו-זמנית מידע אודות מספר מוצרים (Cross Product Query) הן מורכבות יותר, ודורשות תחכום רב יותר של כלי האחזור ושל המשתמש.



תרשים 6.24: ריבוי של טבלאות עובדות.

סכמות פתיתי שלג (Snowflake Schema)

במקרים מסוימים, שימוש בסכמת כוכב לא מנורמלת יוצר בעיות הנובעות מרמת כפילות הנתונים הגבוהה. לדוגמה, אם בכל רמה היררכית יש לנהל מספר רב של תכונות, יכיל המבנה הלא מנורמל רמה גבוהה מאוד של כפילות נתונים. לכן הוא גם יתפוס נפח אחסון רב מאוד. מאידך, נירמול רגיל של היררכיית המימדים גורם למבנה מורכב מאוד, שקשה להבינו ולהשתמש בו. התרשים הבא מציג את מבנה הכוכב, כאשר כל ההיררכיות שלו במצב מנורמל.

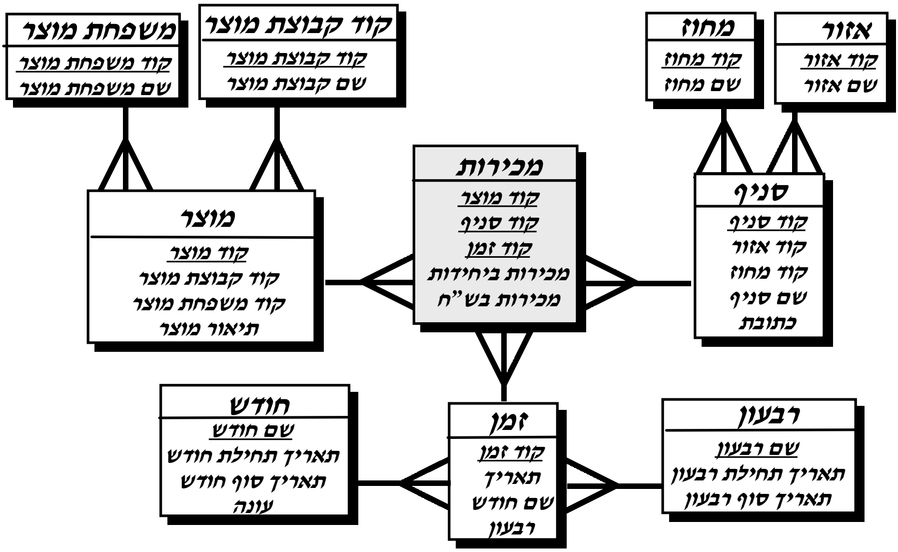


תרשים 6.25: מבנה כוכב עם מימדים מנורמלים.

כפי שניתן לראות, נירמול מלא של המימדים גורם למבנה כוכב מורכב, ובאופן טבעי גם לביצועים לא טובים של השאילתות. השאילתה הבאה מציגה את סך כל המכירות במחוז, בתקופה מסוימת.

```
1.  SELECT      DISTRICT_NAME, SUM (SALES_AMOUNT)
2.  FROM        SALES S, BRANCHES B, AREAS A, DISTRICTS D, PERIODS T
3.  WHERE        T.MONTH = 'February' AND
4.              D.DISTRICT_NAME = 'North' AND
5.              P.PRODUCT_ID = 'ABC-1345' AND
6.              S.BRANCH_ID = B.BRANCH_ID AND
7.              B.AREA_ID = A.AREA_ID AND
8.              A.DISTRICT_ID = D.DISTRICT_ID AND
9.              T.PERIOD_ID = S.PERIOD_ID
10.  GROUP BY  DISTRICT_NAME
11. ORDER BY  DISTRICT_NAME
```

פתרון ביניים מסוים הוא מבנה **סכמת פתיתי שלג**. סכמה זו דומה בעיקרון לסכמת כוכב, למעט העובדה שסביב חלק מהמימדים מופיעות טבלאות נוספות, המייצגות את היררכיות המימדים. להבדיל מהמבנה ההיררכי הרגיל, במקרה זה יוצרים קשר ישיר בין המימד לבין כל ההיררכיות שלו. בתרשים הבא מופיע מבנה פתיתי שלג סביב מימדי הסניפים, המוצרים והזמן. ההחלטה מהם המימדים שיישארו במבנה לא מנורמל, ואילו מימדים ינוהלו במבנה פתיתי שלג, מבוססת על רמת כפילות הנתונים אותה נהיה מוכנים לנהל בבסיס הנתונים, לעומת הפשטות והביצועים של מבנה הנתונים.



תרשים 6.26: מבנה סכמת פתיתי שלג.

במבנה זה ההיררכיה אינה מנוהלת באופן ישיר, אלא עוטפת את המימד. בדרך זו מתקבל מבנה פשוט יותר. מבנה זה עדיין מורכב יותר להבנה, יחסית למבנה הכוכב הלא מנורמל הרגיל, אולם הוא פשוט יותר להבנה לעומת מבנה כוכב עם מימדים מנורמלים. הביצועים במקרה זה פחות טובים מאלו של מבנה הכוכב הלא מנורמל. הגורם לכך הוא הצורך בפעולות צירוף נוספות. נציג כאן את מבנה השאילתה הקודמת, כפי שהיא נראית בסכמת פתיתי שלג. מכיון שקוד המחוז מופיע ישירות בטבלת המימד של הסניף, אין צורך לבצע צירוף עם טבלת האזורים.

```
1.  SELECT      DISTRICT_NAME, SUM (SALES_AMOUNT)
2.  FROM        SALES S, BRANCHES B, DISTRICTS D, PERIODS T
3.  WHERE        T.MONTH = 'February' AND
4.              D.DISTRICT_NAME = 'North' AND
5.              P.PRODUCT_ID = 'ABC-1345' AND
6.              S.BRANCH_ID = B.BRANCH_ID AND
7.              A.DISTRICT_ID = D.DISTRICT_ID AND
8.              T.PERIOD_ID = S.PERIOD_ID
9.  GROUP BY  DISTRICT_NAME
10. ORDER BY  DISTRICT_NAME
```

ניהול שינויים (Change Management)

מדוע יש לנהל שינויים

אחד המאפיינים העיקריים של מחסן הנתונים היא העומק ההיסטורי שלו. כלומר, הצורך לנהל את הנתונים לאורך זמן רב, ממספר חודשים עד מספר שנים. אחד האתגרים העיקריים הנובעים מעובדה זו היא שהארגון משתנה במשך הזמן. כתוצאה מכך, לאורך הזמן משתנה גם מבנה הנתונים במערכות התפעוליות. לדוגמה, הארגון יכול לשנות את המבנה הארגוני, תמהיל המוצרים שהוא מציע ללקוחותיו משתנה, נתוניו של מימד מסוים משתנים (למשל, שינוי במצב המשפחתי של הלקוח) וכדומה.

המערכות התפעוליות מתמודדות עם שינויים אלה בקלות, מכיון שהן צריכות לשקף בעיקר את המצב העדכני האחרון, והן שמות דגש קטן יחסית להיסטוריה. לעומת המערכות התפעוליות, מחסן הנתונים צריך להתמודד עם שינויים אלה ולשקף נכונה את המצב על ציר הזמן. לדוגמה, אם רוצים להשוות בין כמות שיחות הטלפון של לקוחות פרטיים, שמצבם המשפחתי הוא רווק, לעומת כאלה שמצבם המשפחתי הוא נשוי, על מחסן הנתונים לדעת מתי חל שינוי במצבו המשפחתי של הלקוח, ולשייך את השיחות בהתאם. דבר זה מוסיף רמת סיבוכיות נוספת לעיצוב מודל הנתונים של מחסן הנתונים.

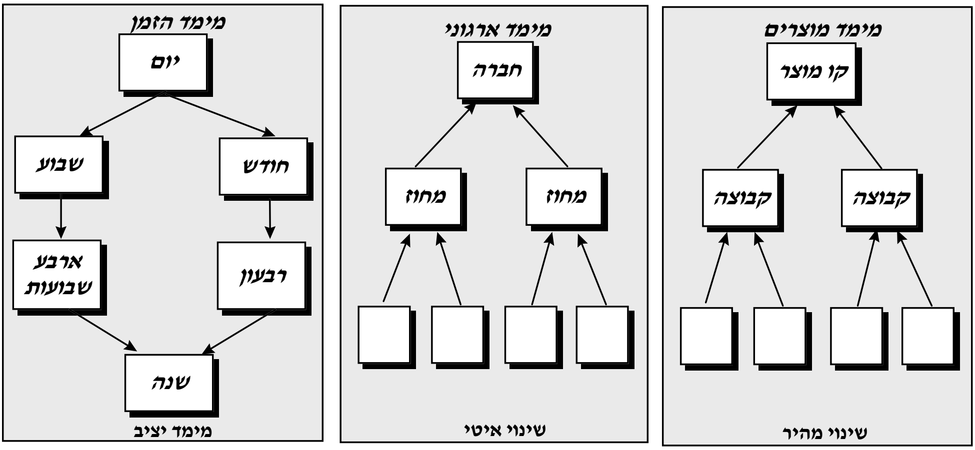
אפיון קצב השינויים

בעיית ניהול השינויים מתמקדת בעיקר בטבלאות המימדים, ולא בטבלת העובדות. הסיבה לכך היא שטבלת העובדות מייצגת אירוע בזמן (שיחת טלפון, פעולה בנקאית, רכישת מוצר וכדומה), ולכן הוא קשור למימדים הנכונים באותה עת. הבעיה היא במימדים המשתנים לאורך הזמן - מצבו המשפחתי של הלקוח היה "רווק" בנקודת זמן מסוימת והשתנה ל"נשוי" בנקודת זמן אחרת. או, למשל, סניף מסוים היה שייך למחוז המרכז בנקודת זמן מסוימת וכתוצאה משינוי במבנה הארגוני הוא שייך למחוז הצפון החל מנקודת זמן אחרת. לכאורה, אחד הפתרונות האפשריים לכך הוא לרשום בטבלת העובדות את מצבו של כל מימד, כפי שהוא היה נכון בעת האירוע. במציאות, פתרון מסוג זה אינו אפשרי, מכיון שהוא גורם לכפילות נתונים בלתי נסבלת כתוצאה מהצורך לנהל את כל נתוני המימדים בכל אירוע.

לפני שנציע מספר פתרונות להתמודד עם שינויים אלה, נאפיין את המימדים לפי קצב השינויים. כפי שמייד נראה קיימים מימדים יציבים דיים, ולעומתם מימדים בהם קצב השינויים הוא גבוה יחסית. מקובל לאפיין את השינויים לשלוש קטגוריות עיקריות:

- ❖ **מימד יציב** (Stable Dimension) - מימד שאינו משתנה, ונשאר יציב לאורך זמן. למשל, מימד הזמן הוא מימד יציב, מכיון שהיררכיות הזמן הן קבועות.
- ❖ **מימד המשתנה באיטיות** (Slowly Changing Dimension) - מימד המשתנה בקצב איטי, יחסית לאורך הזמן. למשל, סביר להניח שמימד המבנה הארגוני ישתנה אולם בקצב איטי - למשל כל שנה. לא סביר להניח שמימד זה ישתנה, למשל, כל חודש.

❖ **מימד המשתנה במהירות** (Fast Changing Dimension) - מימד המשתנה בקצב מהיר, יחסית לאורך הזמן. למשל, סביר להניח שחברת טלקומוניקציה תציע כמעט מדי חודש שירותים חדשים, בנק יציע כל חודש מוצרים בנקאיים חדשים וכדומה.



תרשים 6.27: סיווג השינויים לפי הקצב.

ברור שמימד יציב אינו דורש טיפול בשינויים. לעומתו, מימדים המשתנים באיטיות או במהירות דורשים טיפול מיוחד.

טכניקות לטיפול בשינויים במודל הנתונים

נסקור כאן אוסף של טכניקות שונות, שמטרתן לאפשר את שינוי מודל הנתונים. השינויים ינבעו כתוצאה משינויים במערכות התפעוליות, או שינויים בדרישות משתמשי מחסן הנתונים.

הוספת תכונות חדשות לטבלאות העובדות או המימדים

תוך כדי תפעול מחסן הנתונים מתעורר הצורך בהוספת תכונות חדשות, הן לטבלת העובדות והן לטבלאות המימדים. שינויים אלה יכולים לנבוע כתוצאה משינויים בתהליכי העבודה, כפי שהם משתקפים במערכות התפעוליות, או כתוצאה מדרישות חדשות של משתמשי מחסן הנתונים. סוג זה של שינויים מטופל בקלות רבה מאוד, בשל יכולת מערכות RDBMS להוסיף עמודות חדשות לטבלאות, אפילו תוך כדי פעולה. משמעות הדבר היא, שאין צורך להפסיק את פעולת מחסן הנתונים כדי לבצע את השינוי. הוספת העמודות החדשות לטבלה תתבצע תוך שימוש בפקודת Alter Table בשפת SQL.

הוספת מימד חדש למודל

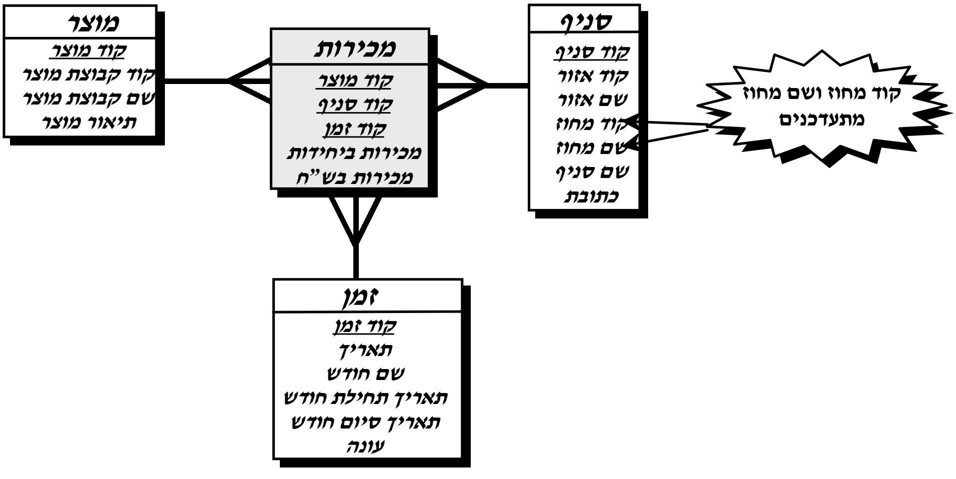
הוספת מימד חדש מתבצע על ידי הגדרת הטבלה החדשה והוספת נתונים לתוכה. הבעיה העיקרית היא הקישור שלה לטבלת העובדות. הקשר הלוגי בין טבלת המימד החדשה לטבלת העובדות, מתבצע על ידי הוספת מפתח זר חדש בטבלת העובדות. בדרך כלל, מפתח טבלת העובדות הוא מפתח המורכב מהמפתחות הזרים. משמעות הדבר היא שיש להוסיף עמודה חדשה, להגדירה כחלק מהמפתח העיקרי, להגדיר אותה כמפתח זר, לעדכן את האינדקס על המפתח העיקרי ולבנות אינדקס חדש על פי המפתח הזר. ביצוע פעולות אלו הוא מורכב יחסית, ומחייב עצירת פעולת בסיס הנתונים, שינוי הגדרת הטבלה וטעינתה מחדש. מכיון שטבלת העובדות גדולה, פעילות זו צורכת משאבי מחשב רבים ועלולה להימשך זמן רב.

ניהול שינויים בהיררכיות של המימדים

נציג כאן מספר חלופות לטיפול בשינויים, הנובעים משינויים בהיררכיות המימדים.

ניהול ערכים עדכניים בלבד (Overwriting Old Values)

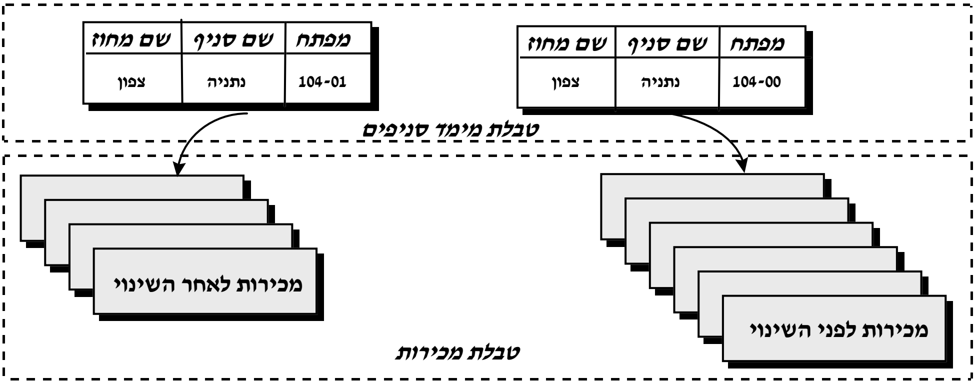
מכיון שהטיפול בשינויים לאורך זמן מכניס סיבוכיות למודל הנתונים, קיימים מצבים בהם ניתן להחליט כי מוותרים על הייצוג ההיסטורי הנכון, על חשבון פשטות מבנה הנתונים. בשיטה זו מעדכנים את הנתון הרלוונטי, ולא שומרים היסטוריה. לדוגמה, אם כתוצאה משינוי במבנה הארגוני, עבר סניף מסוים ממחוז אחד לשני, נעדכן בטבלת הסניף את קוד המחוז ואת שם המחוז, כך שרק המבנה הארגוני החדש ישתקף במחסן הנתונים. ברור, שצורת טיפול זו היא פשוטה מאוד, אולם היא גורמת לייצוג לא נכון של הנתונים על ציר הזמן.



תרשים 6.28: עדכון ערכים.

בניית רשומת מימד חדשה

באותם מקרים בהם להיסטוריה יש חשיבות, ניתן ברגע שאחד מנתוני המימד העיקריים משתנה, לבנות רשומת מימד חדשה. לדוגמה, כתוצאה משינוי במספר המחוזות ובפריסה הארצית של כל מחוז, יש להעביר סניף מסוים ממחוז המרכז למחוז הצפון. כדי שלא לעוות את המידע המסופק ממחסן הנתונים, יש חשיבות להציג את מצב המכירות הנכון, כפי שבוצעו על ידי סניפי המחוז. במקרה זה, ניתן ליצור רשומה חדשה עבור סניף, ובה נתוני המחוז המעודכנים. לרשומה החדשה ניתן להקצות מפתח מלאכותי חדש לחלוטין, או לחילופין להוסיף למפתח המלאכותי הקודם סיומת מתאימה.

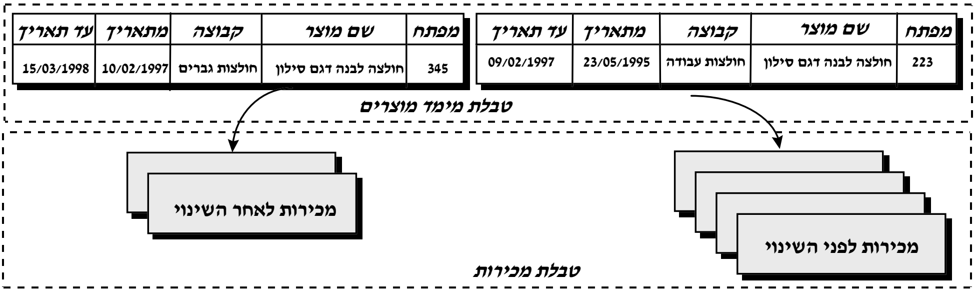


תרשים 6.29: יצירת רשומת מימד חדשה.

בתרשים 6.29 רואים שיצירת הרשומה החדשה משמרת בצורה נכונה את המידע. כל המכירות שבוצעו על ידי הסניף, כל עוד היה שייך למחוז המרכז, ייזקפו למחוז המרכז. כל המכירות שבוצעו על ידי הסניף לאחר שהועבר למחוז הצפון, ישויכו למחוז הצפון. פתרון זה אינו מחייב את המשתמשים לדעת את התאריכים בהם בוצעו השינויים הארגוניים, מפני שהשיוך מתבצע באופן טבעי לחלוטין. כאשר משתמש שואל שאילתה לגבי סך כל מכירות מחוז המרכז, הוא יקבל את מכירות סניף נתניה, רק לתקופה שהסניף היה שייך למחוז זה. שיטה זו מחייבת סיבוך מסוים של תהליך הטעינה למחסן הנתונים, כדי לאתר את השינויים שחלו בטבלת המימד ואת יצירת הרשומה החדשה בהתאם.

נתבונן בדוגמה נוספת לשינויים במימד - שינויים במימד המוצר. בדרך כלל, מקובל לשייך כל מוצר לקבוצת מוצר מסוימת. לדוגמה, מוצר מסוים יכול להשתייך לקבוצת **חולצות גברים** ואילו מוצר אחר יכול להשתייך לקבוצת **חולצות נשים**. רשתות השיווק מבצעות מעת לעת שינויים בשיוך המוצרים לקבוצות המוצר. למשל, יכול להיות שחולצת עבודה השתייכה עד כה לקבוצה מיוחדת של **חולצות עבודה**, ואילו עכשיו מבקשת הרשת לשייך אותה לקבוצה חולצות גברים. הבעיה מתעוררת במידה ויש לנהל את ההיסטוריה של השינויים, כדי להיות מסוגלים לבצע השוואות במכירות בתקופות שונות. אם נרצה להשוות את מכירות חולצות גברים השנה לעומת שנה קודמת, יש לשים לב שבשנה קודמת חולצות העבודה כלל לא היו שייכות לקבוצה חולצות גברים. לכן, כדי לקבל השוואה אמיתית אין להתחשב במכירות חולצות העבודה בשנה קודמת. הדרך להתמודד

עם בעיה זו היא על ידי פתיחת רשומת מימד חדשה עבור חולצת העבודה, כדי לציין את השינוי שחל בקבוצת המוצר. בכל רשומת מימד ננהל גם את תאריכי התוקף שלה, כלומר מאיזה תאריך ועד איזה תאריך המידע המופיע ברשומה נכון.



תרשים 6.30: ניהול רשומות שינויים עם תאריכי תוקף.

תרשים 6.30 מציג את שתי רשומות המוצר בהן מופיע השינוי שחל בהשתייכות לקבוצות המוצר, יחד עם תאריכי תוקף השינוי. אם נרצה להשוות את המכירות של שנת 1998 לעומת אלו של שנת 1997, יש לבצע שתי שאילתות שונות.

```
1. SELECT    SUM (SALES_AMOUNT)
2. FROM      SALES S, PRODUCTS P
3. WHERE     P.PRODUCT_GROUP = 'Menswear' AND
4.           P.START_DATE >= 1/01/1997 AND
5.           P.END_DATE <= 31/12/1997 AND
6.           S.PRODUCT_ID = P.PRODUCT_ID
```

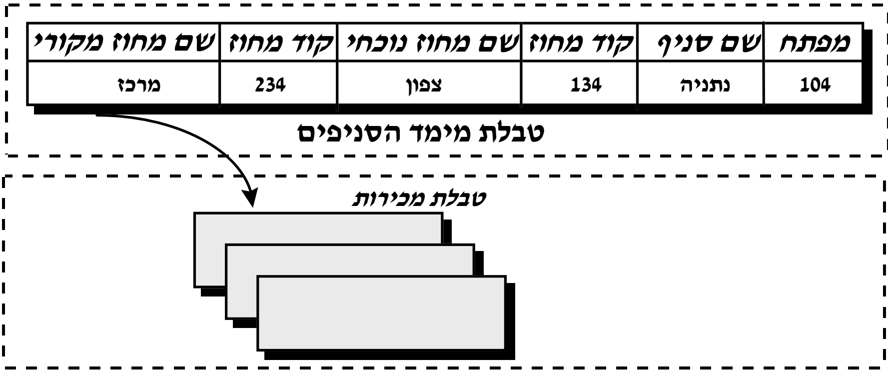
את התוצאה המתקבלת משאילתה זו נשווה לתוצאותיה של שאילתה בה התאריכים מתייחסים לשנת 1998. אם נרצה להשוות את מכירות חולצות הגברים בעונת החורף של שנת 1997, עם עונת החורף של שנת 1998, נשתמש בשאילתה הבאה:

```
1. SELECT    SUM (SALES_AMOUNT)
2. FROM      SALES S, PERIODS T
3. WHERE     P.PRODUCT_GROUP = 'Menswear' AND
4.           T.SEASON = 'Winter 1997' AND
5.           T.TIME_DATE BETWEEN P.START_DATE AND P.END_DATE AND
6.           S.PRODUCT_ID = P.PRODUCT_ID
```

שאילתה זו מחשבת את מכירות כל המוצרים שהשתייכו לקבוצה **חולצות גברים** בחורף 1997. שים לב למשפט 5, המבצע צירוף בין מימד הזמן לבין המכירות, בהתבסס על טווח תאריכי תוקף המוצר.

ניהול מצב עדכני ומצב נוסף

בטכניקה זו מנוהלת ההיסטוריה באופן חלקי בלבד. ברשומת המימד נשמור את הערך הנוכחי ואת הערך הקודם. לדוגמה, אם השיוך הארגוני של הסניף שונה, נשמור ברשומה את קוד ושם המחוז החדש, וגם את קוד ושם המחוז הקודם.



תרשים 6.31: ניהול ערך נוכחי וקודם.

על המעצב להחליט מה לעשות אם יהיה שינוי נוסף - האם לשמור את המצב המקורי, או את המצב הקודם. בדוגמה שהוצגה קודם לכן, אם מתבצע שינוי נוסף והסניף עובר למחוז עמק חפר, יש להחליט האם במחוז הקודם יישמר המחוז המקורי, כלומר מחוז המרכז, או לחילופין המחוז הקודם, כלומר מחוז הצפון. עקרונית ניתן להצמיד לרשומה גם תאריך המצביע על מועד ביצוע השינוי בשיוך הסניף למחוז.

ניתן להתייחס אל טכניקה זו כאל פשרה מסוימת בין שתי הטכניקות הקודמות - מצד אחד לא מתעלמים לחלוטין מהשינויים כמו בשיטה הראשונה, ומאידך לא מנהלים את כל היסטוריית השינויים כמו השיטה השנייה. טכניקה זו קלה יותר ליישום, מפני שאין צורך ביצירת רשומה נוספת והקצאת מפתח חדש. החסרון המרכזי שלה נובע מאיבוד הרצף ההיסטורי והיכולת לשייך במדויק את המכירות למצב האמיתי בכל עת.

ניהול סיכומים (Aggregate Management)

הצורך בניהול סיכומים

סיכומים מהווים מרכיב חשוב בתפיסת מחסן הנתונים. מספר רב מאוד של שאילתות המופעלות בסביבת מחסן הנתונים מתייחסות לסיכומים ברמות שונות - מה היה סך כל התביעות שקיבלנו בשנה אחרונה לסוג מסוים של לקוחות חברת הביטוח, מי הם סניפי הבנק בהם סך כל ההפקדות לתוכניות חסכון מסוג מסוים היו הגבוהות ביותר בחודשיים האחרונים, מי הם יעדי השיחות בחברת טלקומוניקציה שסך כל דקות השיחה אליהם היה הנמוך ביותר ברבעון האחרון, וכדומה. ניתן לבצע סיכומים אלה תוך כדי ביצוע השאילתה, מתוך הנחה שכל הנתונים הדרושים לביצוע הסיכומים מנוהלים במחסן הנתונים. הבעיה היא זמן העיבוד הנדרש לאחזור כל השורות וביצוע הסיכומים. ניתן להשיג שיפור דרמטי בביצוע השאילתות אם מחסן הנתונים יכיל סיכומים מחושבים מראש (Precalculated Aggregates).

החשיבות בניהול הסיכומים נובעת משיפור הביצועים המשמעותי אותו ניתן להשיג. שיפור זה בא על חשבון גידול בנפח אחסון הנתונים וסיבוכיות נוספת, הנובעת מתחזוקת הסיכומים. הגידול בנפח הנתונים והסיבוכיות הנוספת הם כתוצאה מכך שעם כל טעינה חדשה של נתונים למחסן הנתונים יש לעדכן, או לחשב מחדש, את כל הסיכומים הרלוונטיים.

השאלה היא, כיצד ניתן לסכם את הנתונים באופן משמעותי, או מהן הדרכים שתשפרנה את השאילתות הנפוצות. אם נתבונן במימדים המנוהלים במחסן הנתונים נראה את ההירכיות הפנימיות: מוצר, קבוצת מוצר, קו מוצר או סניף, מחוז, אזור או יום, שבוע, חודש, רבעון וכדומה. אלה הם המועמדים הטבעיים לניהול סיכומים ברמות השונות של ההירכיה (Aggregate Rollup).

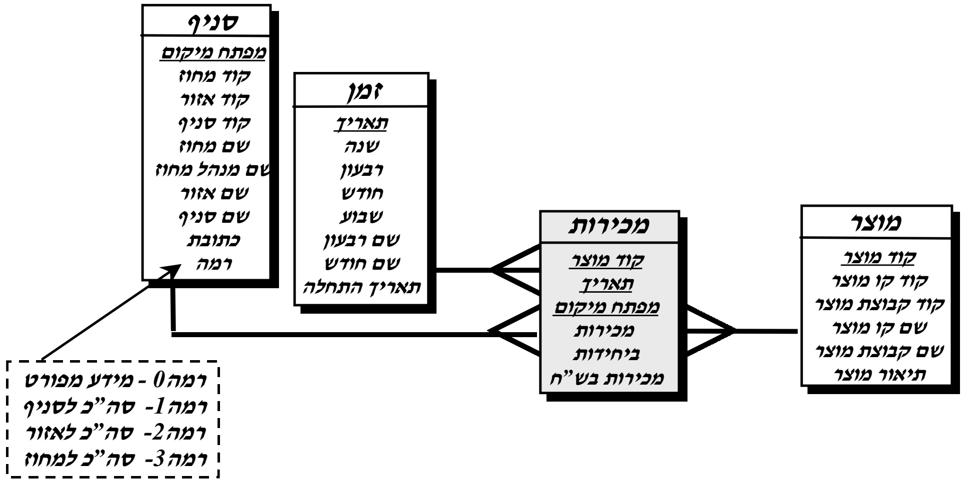
ניתן לנהל סיכומים בכל מימד בנפרד, עבור רמות הירכיות שונות של אותו מימד ועבור צירופים מסוימים של מימדים והירכיות בתוך המימדים. מכיון שכמות הצירופים האפשריים גדולה מאוד, אין כל הגיון בחישוב מראש של כל הסיכומים האפשריים. את הסיכומים יש לבנות מראש עבור השאילתות הנמצאות בשימוש בתדירות גבוהה. קשה מאוד לדעת מראש מה תהיינה השאילתות הנפוצות ביותר. לכן מקובל לבנות תחילה כמות מצומצמת מאוד של סיכומים, לעקוב אחר השאילתות ולבנות סיכומים המבוססים על סטטיסטיקות של השאילתות. בהתאם לסטטיסטיקות אלו ניתן להוסיף סיכומים חדשים, או למחוק סיכומים שהשימוש בהם נדיר.

טכניקות לניהול סיכומים

בעיקרון קיימות שתי שיטות עיקריות לניהול סיכומים: ניהול הסיכומים בטבלה נפרדת, או ניהול הנתונים יחד עם שאר הנתונים בטבלת העובדות. למרות ששתי השיטות דומות, מבחינת נפח הדיסק הנוסף הנדרש לניהול הסיכומים, הן נבדלות באופן משמעותי בצורת השימוש, נוחות הניהול וצורת העדכון שלהן.

ניהול הסיכומים בטבלת העובדות

מטרת שיטה זו לנצל את מודל הנתונים הקיים כדי לנהל את הסיכומים, מבלי להוסיף טבלאות חדשות. הסיכומים עצמם מנוהלים בטבלת העובדות, בתור שורות מיוחדות. נשתמש בדוגמה המופיעה בתרשים הבא כדי להסביר את השיטה. נניח שאנו רוצים לנהל את סך כל המכירות ברמת המימד הגיאוגרפי, כלומר סניף, אזור ומחוז. משמעות הדבר היא שיהיה צורך לנהל בטבלת המכירות מספר שורות מיוחדות - שורה אחת עבור סך כל המכירות לכל סניף, שורה אחת עבור סך כל המכירות לכל אזור ושורה אחת עבור סך כל המכירות לכל מחוז. לדוגמה, אם יש 70 סניפים, 10 אזורים ו-4 מחוזות נצטרך לנהל 84 שורות נוספות בטבלת העובדות. לכל אחת מ-84 שורות אלו נצטרך לקבוע מפתח מיוחד. נוסיף לטבלת המימד של הסניפים 84 שורות נוספות, ולהן ניתן מפתח חד משמעי. כמו כן, נוסיף עמודה מיוחדת לטבלת הסניפים - רמת הסיכום. עמודה זו תקבל אחד מארבעת הערכים האפשריים - 1 יציין שזו שורה של מידע מפורט, 2 יציין שזו שורת סיכום ברמת סניף, 3 יציין שזו רמת סיכום ברמת האזור ו-4 יציין שזו רמת סיכום ברמת המחוז.



תרשים 6.32: מבנה כוכב עם סיכום ברמת סניף, אזור ומחוז.

בתרשים הבא נדגים את תוכן טבלת העובדות וטבלת הסניפים. מטבלת הסניפים ניתן לראות שסניפים S1, S2, ו-S3 שייכים כולם לאזור R1 במחוז D1. סניפים S4 ו-S5 שייכים לאזור R2 במחוז D1. המפתח K6 בטבלת הסניפים מציין שזוהי שורת סיכום עבור כל הסניפים השייכים לאזור R1 במחוז D1. בדומה, המפתח K7 מציין שזוהי שורת הסיכום של כל הסניפים השייכים לאזור R2 במחוז D1. ואכן, אם נסתכל בטבלת העובדות על השורה שהמפתח שלה הוא K6, נראה שהעמודה **מכירות בש"ח** מכילה את סיכום המכירות של סניף S1. בדומה, השורה עם מפתח K7 מכילה את סיכום המכירות של סניף S2. השורה עם מפתח K8 מכילה את סיכום המכירות של כל הסניפים השייכים לאזור R1, וכן הלאה.

טבלת עובדות

מכירות בש"ח	מפתח זמן	מפתח מוצר	מפתח סניף
100	T1	P1	K1
150	T2	P1	K1
200	T1	P2	K1
50	T1	P2	K2
60	T2	P2	K2
450	T2	P3	K2
110	T2	P4	K3
560	T3	P5	K3
120	T1	P1	K4
300	T1	P3	K4
350	T4	P6	K5
450	T4	P6	K5
450	Null	Null	K6
560	Null	Null	K7
1680	Null	Null	K8
1220	Null	Null	K9
2900	Null	Null	K10

טבלת מימד גיאוגרפי

רמת סיכום	קוד סניף	מפתח אזור	מפתח מחוז	מפתח סניף
0	S1	R1	D1	K1
0	S2	R1	D1	K2
0	S3	R1	D1	K3
0	S4	R2	D1	K4
0	S5	R2	D1	K5
1	S1	R1	D1	K6
1	S2	R2	D1	K7
2	Null	R1	D1	K8
2	Null	R2	D1	K9
3	Null	Null	D1	K10

תרשים 6.33: תוכן טבלת עובדות וסניפים עם סיכומים.

החסרון העיקרי של שיטה זו הוא שכל שאילתה צריכה להכיל התייחסות לעמודה המציינת את רמת הסיכום. התעלמות מעמודה זו יכולה להביא לתשובה שגויה של השאילתה. גם שאילתה המבקשת לראות את השורות הפרטניות חייבת להגביל את השאילתה לשורות עם רמת סיכום 0, אחרת גם שורות סיכומיות יצטרפו לתוצאה. בהמשך, מופיעה שאילתה המבקשת להציג את סך כל המכירות עבור כל אחד משלושת המחוזות צפון, מרכז ודרום.

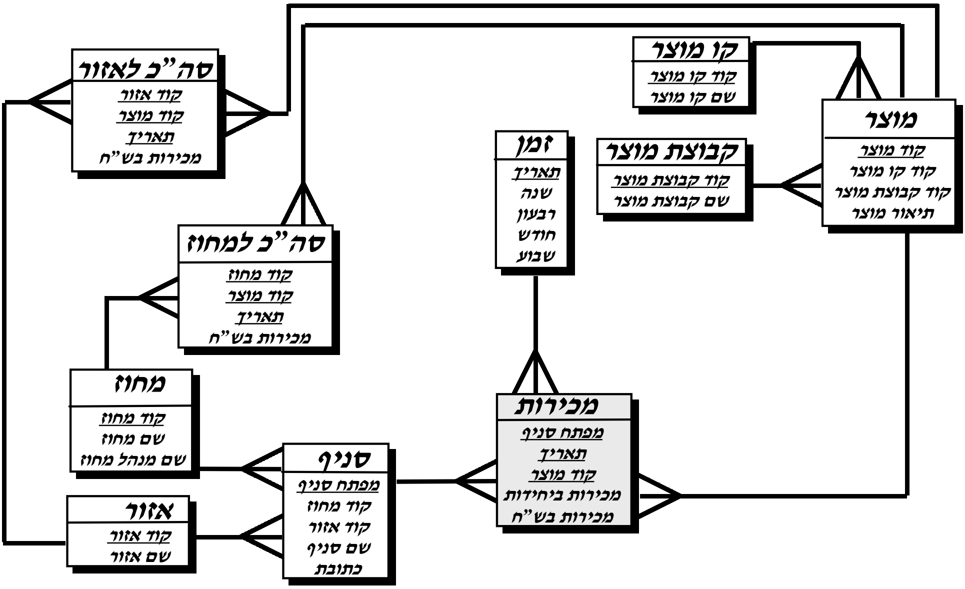
```
1. SELECT    DISTRICT_NAME, SALES_AMOUNT
2. FROM      SALES
3. WHERE     STORE_KEY IN
4.           (SELECT STORE_KEY
5.           FROM STORES
6.           WHERE DISTRICT_NAME IN ('North', 'Central', 'South') AND
7.            LEVEL = 3)
```

מכיון שמציין רמת הסיכום מופיע רק בטבלת הסניפים יש תחילה לשלוף, באמצעות תת שאילתה, את מפתחות רשומות הסיכום ברמת המחוז (כלומר, אלה בהם מופיעה רמת סיכום 3) ולמסור מפתחות אלה לשאילתה הראשית, כדי להציג את סך כל המכירות. שים לב שלמרות ששאילתה זו מציגה סיכומים, היא אינה משתמשת בפונקציה Sum של שפת SQL. זאת מכיון שהסיכום כבר מחושב ומנוהל בטבלת העובדות.

בנוסף לחסרון המוזכר, שיטה זו היא קשיחה ומחייבת קביעה מראש של רמות הסיכום המבוקשות. כל שינוי ברמות הסיכום הוא מורכב, ודורש טיפול בטבלת העובדות (הטבלה הגדולה ביותר, והטיפול בה הוא המורכב ביותר).

ניהול טבלאות סיכומים נפרדות

בשיטה זו מנוהלים הסיכומים בטבלאות נפרדות וייעודיות. בתרשים הבא מופיעה דוגמה לסכמת פתיתי שלג, המכילה גם שתי טבלאות סיכומיות - סיכום מכירות עבור האזור ועבור המחוז. היתרון העיקרי של שיטה זו נובע מההפרדה הברורה בין טבלת העובדות לבין טבלאות הסיכומים. אין צורך להתייחס בכל שאילתה לעמודה המיוחדת של רמת הסיכום, והפנייה לטבלאות הסיכומים מתבצעת רק כאשר השאילתה צריכה להציג סיכומים. שיטת ניהול טבלאות אלו היא פשוטה - מייד עם סיום טעינת נתונים חדשים למחסן הנתונים מתבצעת מחיקת טבלאות הסיכומים ומופעלות פקודות SQL, ליצירה מחדש של טבלת הסיכומים.



תרשים 6.34: סכמת פתיתי שלג עם שתי טבלאות סיכומים.

יתרונות וחסרונות שיטות ניהול הסיכומים

לכל אחת מהשיטות המוזכרות כאן יש יתרונות וחסרונות. הטבלה הבאה מציגה באופן תמציתי את עיקרי ההבדלים בין שתי השיטות.

	שימוש במצייני רמת סיכום	**ניהול טבלאות סיכומים נפרדות**
מספר הטבלאות	מספר הטבלאות המנוהלות אינו משתנה כתוצאה מניהול הסיכומים.	מספר הטבלאות גדל. החלטה לנהל מספר רב של סיכומים יכולה להגדיל

	שימוש במצייני רמת סיכום	ניהול טבלאות סיכומים נפרדות
	הסיכומים מנוהלים במבנה הטבעי של בסיס הנתונים.	באופן משמעותי את מספר הטבלאות.
סיבוך השאילתה	מחייב את המשתמש להבין את מצייני הרמה, ולהשתמש בהם בכל שאילתה. אי שימוש במצייני הרמה עלול להביא להצגת תוצאות לא נכונות.	מחייב את המשתמש להכיר את טבלאות הסיכומים, ולהפנות את השאילתה לטבלאות הסיכומים, או לטבלת העובדות. כיום, קיימים כלים המאפשרים הגדרת טבלאות הסיכום במילון הנתונים, וביצוע המרה אוטומטית של משפטי SQL הפונים לטבלאות העובדות לפני הפנייה לטבלאות הסיכום.
ביצועים	ביצועים טובים פחות, מפני שהסיכומים מנוהלים כחלק מטבלת העובדות הרגילה (שממילא היא הטבלה הגדולה ביותר). אינדקסים של טבלת העובדות מסייעים בשיפור ביצועי השאילתה.	ביצועים טובים, אם צריך רק בנתונים הסיכומים. אם צריך לבצע צירוף עם טבלאות נוספות הביצועים יורדים.
ניהול בסיס הנתונים	פשוט יחסית. אין שינוי לעומת מצב רגיל, מפני שכמות הטבלאות אינה משתנה.	מורכב יותר, מפני שמספר הטבלאות גדול יותר. מבחינת גיבויים/שחזורים - שיטה זו מורכבת יותר.
רגישות לשינויים	גדולה יותר. הוספת סיכומים חדשים יכולה להיות מורכבת למדי, בשל הצורך לגעת בטבלאות העובדות והמימדים. הוספת רמת סיכום בין רמות קיימות, יכולה להיות פעולה מורכבת מאוד.	קטנה יחסית. פעולת הוספת סיכום חדש, או ביטול סיכום קיים, אינה משפיעה על יתר הטבלאות. כל שנדרש הוא לבנות טבלת סיכומים חדשה, או לבטל טבלת סיכומים קיימת.
חישוב הסיכומים	מורכב יותר. הלוגיקה של מצייני הרמה מורכבת יותר, ובדרך כלל מחייבת שימוש בתוכניות מיוחדות.	פשוט יחסית. ניתן להכין את טבלאות הסיכומים באמצעות פקודות SQL פשוטות - מחיקת טבלת הסיכומים והכנת טבלה חדשה.

השיטה המועדפת כיום על מעצבי בסיסי הנתונים בסביבת מחסן נתונים היא ניהול נפרד של טבלאות הסיכומים, בעיקר בשל פשטות השיטה ואי רגישותה לשינויים בדרישות לניהול הסיכומים.

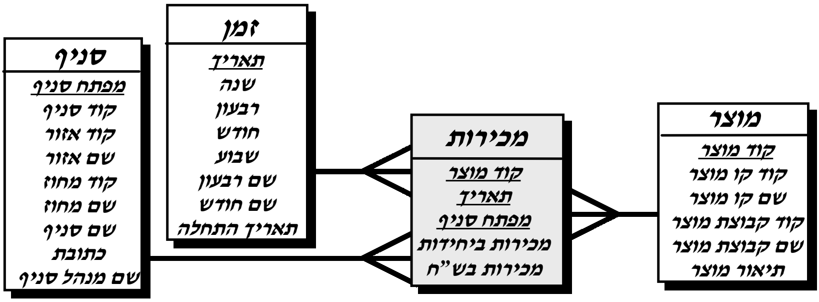
קביעת מספר טבלאות הסיכומים שיש לנהל

במובן מסוים, ניתן להסתכל על טבלאות הסיכומים כהרחבה טבעית של טבלת העובדות. הוספת טבלאות הסיכומים אינה משנה את המבנה הבסיסי של טבלאות העובדות. לכן, סכמה של בסיס הנתונים נשארת במבנה סכמת הכוכב, או פתיתי שלג. כל ההבדל הוא שרשומות מטבלת העובדות מסוכמות, וזאת כדי ליעל את השאילתות.

ההחלטה אילו טבלאות סיכומים ינוהלו, אינה החלטה פשוטה. מאחר והשאילתות המופעלות בסביבת מחסן הנתונים משתנות, יהיה צורך לשנות גם את טבלאות הסיכומים

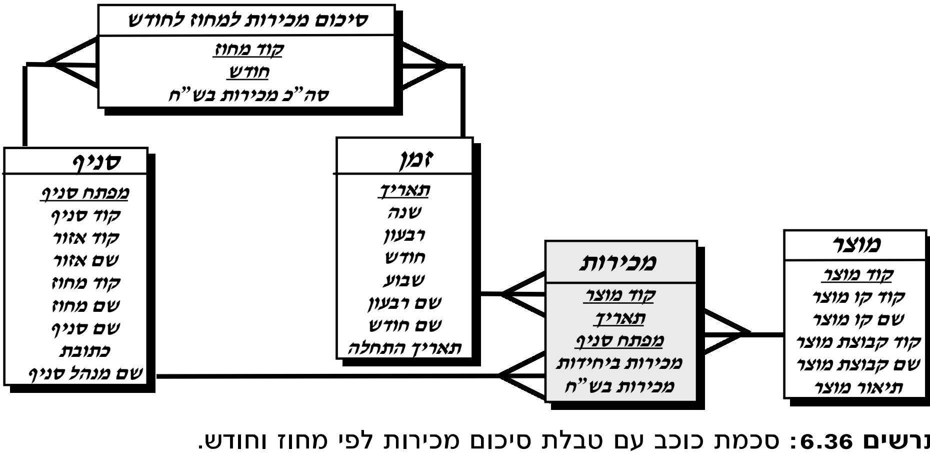
בהתאם. בכל מקרה, יש להגדיר לפחות את האוכלוסיה הראשונית של טבלאות הסיכומים, ומעת לעת להתאימה לדרישות. למרות הנטייה הטבעית להגדיר מספר רב של טבלאות סיכומים, יש לבחון את הצורך בטבלאות הסיכומים. יש לבנות רק את אלו שסביר שיתבקשו על ידי המשתמשים וסביר שיביאו לחסכון משמעותי בזמן העיבוד. מכיון שבדרך כלל קשה לנחש מראש את הסיכומים הדרושים, מומלץ לבנות רק סיכומים שברור שידרשו, ובהמשך לעקוב אחר השימוש במחסן הנתונים ולבנות טבלאות חדשות, או לבטל קיימות, לפי הצורך.

לדוגמה, נתייחס למחסן נתונים העוסק במכירות, ויש בו טבלת עובדות המכילה את סך כל המכירות היומיות לכל מוצר וסניף, ושלוש טבלאות מימדים - מוצרים, סניפים וזמן. בתוך מימד המוצרים קיימת היררכיה של מוצרים וקווי מוצר, ובתוך מימד הסניפים קיימת היררכיה של סניפים ואזורים.



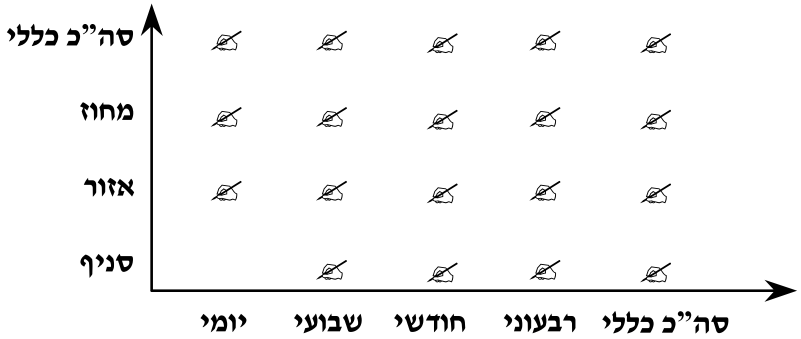
תרשים 6.35: סכמת כוכב עם שלושה מימדים.

בכל אחד מהמימדים קיימות היררכיות במספר רמות. רמת הפירוט המנוהלת בטבלת העובדות היא הרמה היומית, כלומר סך כל המכירות למוצר, לסניף ביום. אם נרצה לנהל סיכומים מעבר לרמת הפירוט המנוהלת בטבלת העובדות, נצטרך לבנות טבלאות סיכומים מתאימות. בתרשים הבא מופיעה טבלת סיכום המכילה את סיכום המכירות למחוז, בחודש. שים לב שהטבלה קשורה רק למימד הזמן ומימד הסניף, ולא למימד המוצרים, מפני שהסיכום מתייחס רק למכירות לפי מחוז וחודש, ולא לפי מוצר.



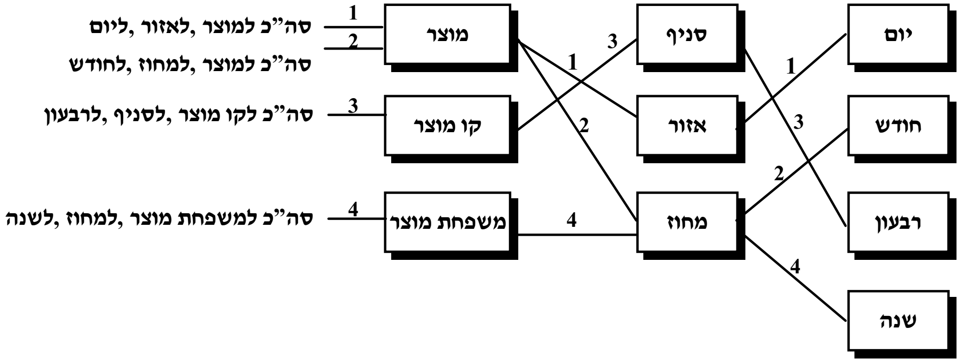
תרשים 6.36: סכמת כוכב עם טבלת סיכום מכירות לפי מחוז וחודש.

ניתן ליצור מספר רב מאוד של צירופים שונים המייצגים רמות סיכום שונות. בעיקרון, יכולנו ליצור טבלת סיכום עבור כל רמה בהירכיית המימד, תוך שמירה על רמת הפירוט בכל יתר המימדים. אם נתייחס רק לשני מימדים, כאשר לכל אחד מהם m ו-n רמות הירכיות בהתאמה, יתקבלו n x m צירופים אפשריים. ככל שמספר המימדים ומספר הרמות ההירכיות בתוך המימד גדלים, כך גם גדל באופן דרמטי מספר הצירופים האפשריים. התרשים הבא מציג דוגמה של הצירופים האפשריים עבור מימד הזמן ומימד הסניף בלבד. שים לב שאין צורך לנהל את הרמה היומית של הסניף, מפני שזו רמת הפירוט המנוהלת בטבלת העובדות.



תרשים 6.37: צירופים אפשריים של סיכומים עבור שני מימדים.

נתבונן בדוגמה נוספת, של צירופים אפשריים עם שלושה מימדים. נציג רק חלק מבין כל הצירופים האפשריים. בתרשים הבא מופיעים מספר צירופים אפשריים וכל צירוף כזה קיבל מספר מזהה. לדוגמה צירוף 1 מייצג סיכום של מוצר לאזור ביום. צירוף 4 מייצג סיכום של משפחת מוצר למחוז בשנה. ניתן ליצור מספר רב של צירופים שונים.



תרשים 6.38: צירופים שונים של רמות סיכום.

מקובל לחלק את טבלאות הסיכומים למספר קטגוריות, בהתאם למספר רמות הביניים המשתתפות בסיכום (מעבר לרמת הפירוט הבסיסית). הרמה הבסיסית, בדוגמה שלנו, היא סך כל המכירות למוצר, לסניף ביום. רמה זו מנוהלת ישירות בטבלת העובדות. אם

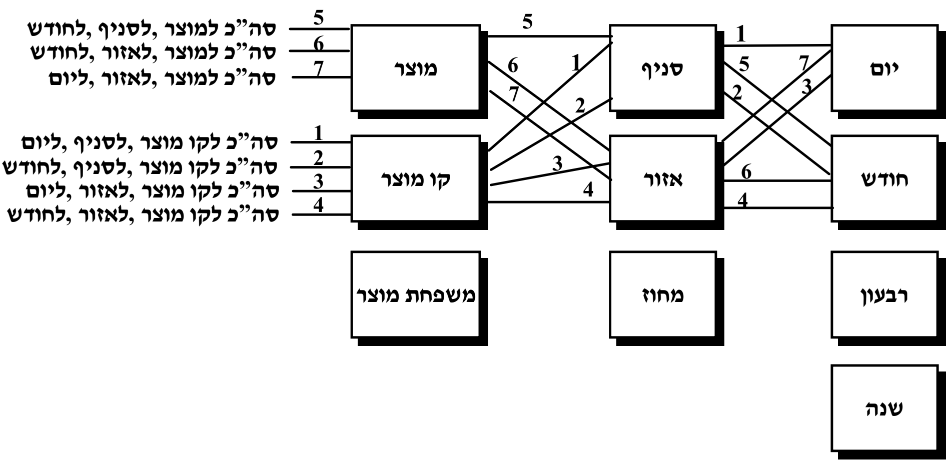
בטבלת הסיכום משתתפת רמת ביניים אחת, נאמר שטבלת הסיכום היא ברמת סיכום ביניים אחת (One Way Aggregate). אם בטבלת הסיכום משתתפות שתי רמות ביניים, נאמר שטבלת הסיכום היא בשתי רמות סיכום ביניים (Two Way Aggregate), וכן הלאה. לדוגמה, בטבלת סיכום לקו מוצר לסניף ליום, קו המוצר הוא ברמת סיכום ביניים. לעומתו, שני המימדים האחרים נמצאים ברמת הבסיס, לכן זהו סיכום ברמה אחת. לעומת זאת, טבלת הסיכום למשפחת מוצר, למחוז לשנה היא ברמה שלוש.

כעת, ננסה לתכנן את מספר טבלאות הסיכום אותן יש לנהל. מספר זה תלוי בדרישות ובשאילתות הטיפוסיות שיפעילו המשתמשים.

נניח, לצורך הדוגמה, שזיהינו את הדרישות הבאות לניהול סיכומים:

- ❖ סיכום המכירות לפי קו מוצר ולפי סניף/אזור במימד הסניפים ולפי יום/חודש במימד הזמן.
- ❖ סיכום המכירות לפי אזור ולפי מוצר/קו מוצר במימד המוצרים ולפי יום/חודש במימד הזמן.
- ❖ סיכום המכירות לפי חודש ולפי מוצר/קו מוצר במימד המוצר ולפי סניף/אזור במימד הסניפים.

התרשים הבא מייצג את כל הצירופים של טבלאות הסיכומים הנדרשות כדי לתמוך בדרישה זו.



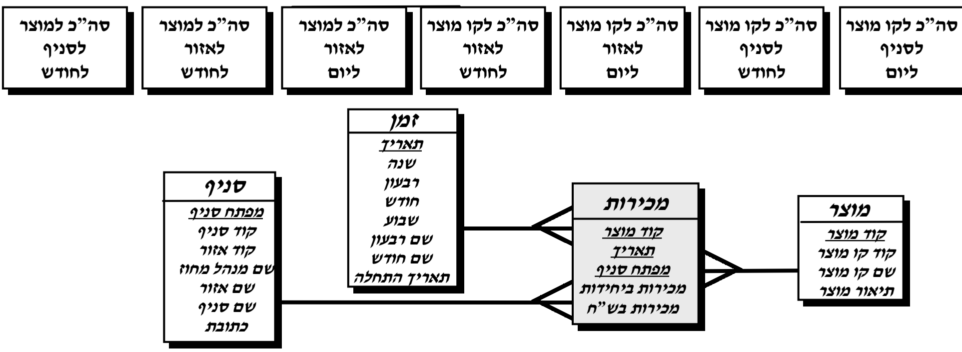
תרשים 6.39: טבלאות סיכומים.

להלן רשימת הטבלאות המתקבלת לפי הקטגוריות של רמות הסיכום:

- ❖ רמה אחת (One Way Aggregate) - סך הכל לקו מוצר לסניף ליום
- ❖ רמה אחת (One Way Aggregate) - סך הכל למוצר לסניף לחודש
- ❖ רמה אחת (One Way Aggregate) - סך הכל למוצר לאזור ליום

- ❖ שתי רמות (Two Way Aggregate) - סך הכל לקו מוצר לאזור ליום
- ❖ שתי רמות (Two Way Aggregate) - סך הכל לקו מוצר לסניף לחודש
- ❖ שתי רמות (Two Way Aggregate) - סך הכל לקו מוצר לאזור ליום
- ❖ שתי רמות (Two Way Aggregate) - סך הכל למוצר לאזור לחודש
- ❖ שלוש רמות (Three Way Aggregate) - סך הכל לקו מוצר לאזור לחודש

התרשים הבא מציג את סכמת בסיס הנתונים, יחד עם טבלאות הסיכומים.



תרשים 6.40: סכמת כוכב עם טבלאות סיכומים.

למרות שמספר הצירופים האפשרי לניהול טבלאות סיכומים יכול להיות גדול מאוד, החדשות הטובות הן שמיותר לנהל טבלאות סיכומים מסוימות, למרות שהן נדרשות על ידי המשתמשים. מאחר וכל טבלת סיכומים דורשת ניהול ותופסת מקום אחסון, מומלץ לבחון היטב אילו טבלאות סיכום ינוהלו בבסיס הנתונים. צמצום מספר טבלאות הסיכומים שננהל יתרחש אם נשים לב לכך שניתן לחשב סיכומים ברמות גבוהות יותר בהיררכיה, מתוך סיכומים ברמות נמוכות יותר. לכן, ייתכן שנעדיף שלא לנהל באופן ממשי טבלאות סיכום מסוימות, אלא לחשב את הסיכומים תוך כדי ביצוע השאילתה. לדוגמה, ניתן לחשב סיכום מכירות רבעוני מתוך הסיכום החודשי, על ידי סיכום 3 שורות בלבד. בדומה, ניתן לחשב את הסיכום השנתי על ידי סיכום 12 שורות של סיכומים חודשיים. לעומת זאת, לא ניתן לחשב את הסיכום הרבעוני או השנתי מתוך סיכומים שבועיים, מכיון ששבוע מסוים יכול להתפרס על פני יותר מרבעון או שנה מסוימת. דוגמה אחרת יכולה להיות ניהול סיכומים לקו מוצר. אם, בממוצע, לכל קו מוצר יש 10 מוצרים, אזי ניתן להגיע בקלות אל סיכום קו מוצר, לסניף, בחודש על ידי סיכום 10 שורות מטבלת סיכום מוצר, לסניף בחודש. לעומת זאת, אם בקו מוצר יש בממוצע אלפי מוצרים, ברור שעדיף לנהל את טבלת הסיכומים באופן ממשי. צורת התבוננות זו יכולה לצמצם באופן משמעותי את מספר טבלאות הסיכומים שננהל בבסיס הנתונים.

קביעת המפתחות והעמודות של טבלת הסיכומים

על המעצב לקבוע לכל טבלת סיכומים את המפתח ואת העמודות הנוספות שינוהלו בטבלה. בדרך כלל מקובל להשתמש בטבלאות הסיכומים במפתחות טבעיים - כגון קוד המוצר, קוד הסניף - ולא במפתחות מלאכותיים. להזכירך, הסיבה העיקרית לשימוש במפתחות מלאכותיים היא כדי לאפשר לנהל בצורה קלה יותר שינויים אפשריים לאורך זמן במפתחות הטבעיים. מכיון שטבלאות הסיכומים ממילא נבנות מחדש כל פעם, אין כל צורך להשתמש במפתחות מלאכותיים.

סך הכל לקו מוצר לסניף ליום

<u>קוד קו מוצר</u>
<u>קוד סניף</u>
<u>תאריך</u>
סה"כ מכירות בש"ח
מונה מספר מכירות

סך הכל לקו מוצר לאזור לחודש

<u>קוד קו מוצר</u>
<u>קוד אזור</u>
<u>חודש</u>
<u>שנה</u>
סה"כ מכירות בש"ח
מונה מספר מכירות

תרשים 6.41: תוכן טבלאות סיכומים.

נשים לב שבדוגמה המופיעה בתרשים 6.41, טבלת הסיכומים מכילה רק את המפתח הזר לטבלת המימדים הרלוונטית. דבר זה מחייב לבצע פעולת צירוף, בין טבלת הסיכומים לבין טבלאות מיוחדות המכילות את התיאורים, בכל גישה לטבלת הסיכומים. כדי לחסוך פעולה זו, ולייעל את השאילתות הסיכומיות, ניתן להחזיק בטבלת הסיכומים את הקודים ואת תיאורי המימדים.

סך הכל לקו מוצר לסניף ליום

<u>קוד קו מוצר</u>
<u>קוד סניף</u>
<u>תאריך</u>
תיאור קו מוצר
שם סניף
סה"כ מכירות בש"ח
מונה מספר מכירות

סך הכל לקו מוצר לאזור לחודש

<u>קוד קו מוצר</u>
<u>קוד אזור</u>
<u>חודש</u>
<u>שנה</u>
תיאור קו מוצר
שם אזור
סה"כ מכירות בש"ח
מונה מספר מכירות

תרשים 6.42: טבלת סיכומים עם תיאורים.

לאחר קביעת מפתחות טבלת הסיכומים, צריך המעצב להחליט איזה סיכומים ינוהלו - למשל, סך הכל מכירות בש״ח, סך הכל מכירות בדולר וכדומה. תנאי בסיסי לניהול הסיכום הוא היות העמודה ברת סיכום (Additive) בהיררכיה, כלומר לסיכום יש משמעות והגיון. בנוסף לסיכומים הברורים, הנובעים מהנתונים המנוהלים בטבלת העובדות, יש לפעמים צורך לנהל גם נתונים מצטברים נוספים כגון ממוצע המכירות לתקופה, המכירה היומית המקסימלית לתקופה, המכירה היומית הנמוכה ביותר לתקופה וכדומה. ניהול

נתונים מצטברים אלה, ישירות בטבלת הסיכומים, יכול לשפר את השאילתות הזקוקות לסוג זה של נתונים.

שים לב שנתון כגון כמות אינו בר סיכום (Semi Additive) בהיררכיות השונות. אין טעם לסכם את כמות המכירות של מוצר מסוים לסה״כ לפי קו המוצר, מאחר ולסיכום זה אין משמעות מפני שמוצרים שונים יכולים להיות בעלי יחידות מידה שונות, כמו מטרים וליטרים.

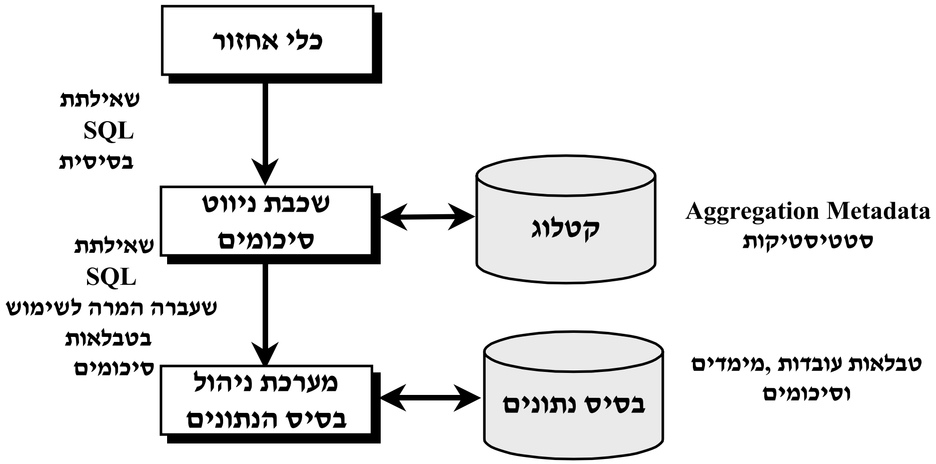


תרשים 6.43: תוכן לדוגמה של טבלאות סיכומים.

זיהוי אוטומטי של טבלאות הסיכומים (Automatic Aggregate Navigation)

כל עוד כלי האחזור אינם מזהים טבלאות אלה כטבלאות סיכומים, על המשתמש לקבוע לאיזו טבלה הוא מבקש להפנות את השאילתה. אם המשתמש אינו מודע לקיומן של טבלאות הסיכומים, הוא יכול להפנות שאילתות ישירות לטבלת העובדות ולבזבז משאבי מחשוב רבים ולסבול מזמני תגובה גרועים. מצב זה אינו טוב, מכיון שהוא מחייב את המשתמש לדעת אודות אוכלוסיית טבלאות הסיכומים המנוהלות בבסיס הנתונים, ועליו לעשות שימוש בידע זה בכל שאילתה.

שיפור משמעותי למצב זה הוא מצב בו כלי האחזור מכיר את הסמנטיקה של הטבלאות ומחליט באופן אוטומטי לאיזו טבלה להפנות את השאילתה. המשתמש כלל אינו מודע לקיומן של טבלאות הסיכומים, ובונה את השאילתה באופן רגיל. כלי האחזור בודק, מול מילון הנתונים, את הטבלאות ומבצע באופן אוטומטי את המרת השאילתה. למעשה, נוצרת כאן שכבה נוספת של תוכנה, המקבלת את שאילתת המשתמש, בודקת אותה מול מילון הנתונים ומבצעת המרה של השאילתה לשאילתה דומה (מבחינה סמנטית), אך בהבדל אחד - השאילתה עושה שימוש בטבלאות הסיכום. בשיטה זו, ניהול טבלאות הסיכומים הוא שקוף, מבחינת המשתמש. הארכיטקטורה המתקבלת מתוארת בתרשים הבא.



תרשים 6.44: ארכיטקטורה של מערכת עם ניווט טבלאות סיכומים.

נציג דוגמה כיצד מבצעת שכבת ניווט הסיכומים את המרת השאילתות. נניח, שבסיס הנתונים כולל, בין היתר, טבלאות סיכומים לקו מוצר, לאזור, בחודש. המשתמש משגר את השאילתה הבאה, המופנית לטבלת העובדות:

```
1.  SELECT      PRODUCT_LINE_NAME, SUM (SALES_AMOUNT)
2.  FROM        SALES S, BRANCHES B, PRODUCTS P, PERIODS T,
3.  WHERE       T.MONTH = 'February' AND
4.              D.DISTRICT_NAME = 'North' AND
5.              P.PRODUCT_LINE_ID = 'Grocery' AND
6.              S.BRANCH_ID = B.BRANCH_ID AND
7.              S.PRODUCT_ID = P.PRODUCT_ID AND
8.              T.PERIOD_ID = S.PERIOD_ID
9.  GROUP BY  PRODUCT_LINE_NAME
10. ORDER BY  PRODUCT_LINE_NAME
```

שכבת ניווט הסיכומים מקבלת את השאילתה ומנתחת אותה. מכיון שהדרישה כאן היא להציג את סך כל המכירות לקו מוצר מסוים - מוצרי מכולת, בחודש פברואר לאזור הצפון ניתן להחליף את השאילתה בשאילתה הבאה:

```
1.  SELECT      PRODUCT_LINE_NAME, SALES_AMOUNT_SUM
2.  FROM        TOTAL_BY_LINE_BY_DISTRICT_BY_MONTH TS,
3.              PRODUCTS P, DISTRICTS D, PERIODS T
3.  WHERE       T.MONTH = 'February' AND
4.              D.DISTRICT_NAME = 'North' AND
5.              P.PRODUCT_LINE_ID = 'Grocery' AND
6.              TS.BRANCH_ID = B.BRANCH_ID AND
7.              TS.PRODUCT_ID = P.PRODUCT_ID AND
8.              TS.PERIOD_ID = T.PERIOD_ID
9.  GROUP BY  PRODUCT_LINE_NAME
10. ORDER BY  PRODUCT_LINE_NAME
```

זמן התגובה של השאילתה החדשה יהיה מהיר מאוד, בשל היכולת להביא את השורה המבוקשת במהירות רבה, תוך שימוש באינדקסים של טבלת הסיכומים. דוגמה זו מבוססת על ההנחה ששם קו המוצר מנוהל בטבלת הסיכומים. אם לא זה המצב, יידרש ביצוע פעולת צירוף עם טבלת קווי המוצר.

כיום, קיימים כבר מספר כלים המבצעים ניווט אוטומטי בטבלאות הסיכומים. ביניהם ניתן למנות את MetaCube של חברת Informix, DecisionSuite של חברת Information Advantage, DSS Server של חברת Micro Strategy. יצרני מערכות RDBMS פועלים להכניס את התמיכה בניווט בטבלאות סיכומים למנוע בסיס הנתונים. לדוגמה חברת Informix שילבה את המוצר MetaCube כאובייקט בגרסת Informix Universal Server שלה.

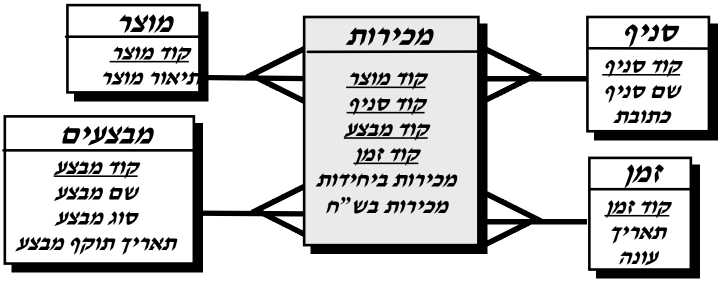
חישוב נפח בסיס הנתונים

אחד הפרמטרים החשובים ביותר של מחסן הנתונים הוא נפח הנתונים הצפוי. פרמטר זה מכתיב את גודל הדיסקים הנדרשים ואת עוצמת השרת. שיטת חישוב נפח בסיס הנתונים פשוטה. השלבים לחישוב נפח בסיס הנתונים הם:

- חישוב נפח טבלאות המימדים, על סמך גודל כל טבלה ומספר השורות הצפויות בה;
- חישוב נפח טבלת העובדות, על סמך רמת הדלילות (Sparcity) (המכתיבה את מספר השורות הצפויות);
- חישוב נפח אינדקסים צפויים;
- חישוב נפח שטחי עבודה.

נפח טבלאות המימדים והעובדות נקרא **נפח הנתונים הגולמיים** (Raw Data), סיכום נפח הנתונים הגולמיים, יחד עם שטחי האינדקסים ושטחי העבודה, נקרא **סך כל נפח מחסן הנתונים**. לעיתים, מקובל להשתמש בכללי אצבע להמרה מנפח האחסון של הנתונים הגולמיים לסך כל נפח מחסן הנתונים. כלל אצבע כזה קובע שנפח שטחי האינדקסים והעבודה המקובלים הם פי 2 עד פי 4 מנפח הנתונים הגולמיים. ככל שנשאיר שטחי עבודה גדולים יותר, יש לצפות לזמני תגובה טובים יותר. זאת, מכיון שמערכת RDBMS משתמשת בשטחים אלה תוך כדי ביצוע השאילתה לאחסון תוצאות ביניים, לביצוע מיונים וכדומה.

הדרך הפשוטה ביותר להדגים את שיטת החישוב היא להשתמש בדוגמה. נתבסס על הדוגמה של סכמת כוכב, המכילה את טבלת העובדות המכילה את סך כל המכירות לחנות, למוצר, ביום, וסביבה מספר טבלאות מימדים.



תרשים 6.45: דוגמת סכמת כוכב לחישוב נפחים.

חישוב נפח טבלאות מימדים

יש לחשב בנפרד את נפחה של כל אחת מטבלאות המימדים, בהתבסס על מספר השורות הצפוי ומספר העמודות בכל טבלה. כדי לפשט את העניין נניח שכל שדה נומרי תופס 4 בתים, וכל שדה טקסט 20 בתים.

❖ **נפח טבלת מוצרים** - בהנחה שיש 40,000 מוצרים שונים, לכל מוצר שדה נומרי אחד ושדה טקסט אחד - יתקבל נפח של 40,000 x (4 + 20) = 960,000 בתים.

❖ **נפח טבלת סניפים** - בהנחה שיש 200 סניפים שונים, לכל סניף שדה נומרי אחד ושני שדות טקסט - יתקבל נפח אחסון של 500 x (4 + 40) = 22,000 בתים.

❖ **נפח טבלת זמן** - בהנחה שנשמור היסטוריה של שנתיים, בכל שנה 365 ימים - יתקבל 2 x 365 x (4+4+20) = 20,440 בתים.

❖ **נפח טבלת מבצעים** - מכיון שסביר להניח שזו טבלה קטנה, נתעלם ממנה לצורך חישוב הנפחים.

סך כל נפח האחסון של טבלאות המימדים יהיה 960,000 + 22,000 + 20,440 = 1,002,440 בתים, כלומר כ- 1GB. זהו הנפח הגולמי. אם נניח יחס של פי 3 להמרת נפח גולמי לנפח כולל, נקבל במקרה זה 4GB.

חישוב נפח טבלת העובדות

בהתחשב בנתונים הקודמים, מספר האפשרויות יכול להגיע לכדי 40,000 x 200 x 730 = 5,840,000,000 רשומות אפשריות בטבלת העובדות, כלומר כמעט 6 מיליארד רשומות. למרבה המזל, לא בכל יום מוכרים כל מוצר בכל חנות, ולכן חישוב זה הוא תיאורטי בלבד. כאן נכנס לתמונה יחס דלילות הנתונים (Data Sparsity), כלומר מהו היחס בין מספר המצבים האמיתיים, למספר המצבים האפשריים. נניח שבכל יום מוכרים רק 2,000 מוצרים בכל חנות, כלומר מקדם הדלילות יהיה 40,000 / 2,000 = 0.05. בבסיס הנתונים נשמור שורה אחת בבסיס הנתונים עבור כל מכירה של מוצר בכל חנות, ולא נשמור שורות עבור מצבים תיאורטיים. בדרך זו, מספר השורות יהיה קטן בהרבה.

נקבל 2,000 x 200 x 730 = 292,000,000 שורות, כלומר כמעט 300 מיליון רשומות. בטבלת העובדות יש שישה שדות נומריים ולכן יתקבל נפח אחסון גולמי: 292,000,000 x 6 x 4 = 7,008,000,000 בתים, כלומר כ- 7GB.

על פי אותו יחס המרה של נפח גולמי לנפח כולל, נקבל 21GB.

פרק 7: כריית נתונים (Data Mining)

מבוא

כריית נתונים היא אחד הנושאים החמים ביותר בתחום מחסני נתונים. כשעסקנו בפרק 3 בנושאי המרחב האנליטי, ראינו שטכנולוגיית כריית הנתונים היא זו המאפשרת לנו להפוך נתונים למידע ולידע על בסיס מודל הגילוי, ולא על בסיס מודל האימות. בשנים האחרונות אנו עדים לגידול מרשים בכמות הנתונים המנוהלים באופן אלקטרוני. יש הקוראים לתופעה זו גם "התפוצצות המידע". עלות יחידות אחסון הנתונים נמצאת במגמת ירידה מתמדת, ולכן התלבטות הארגון לגבי איסוף הנתונים הופכת לפחות ופחות רלוונטית. ההתלבטות החשובה יותר היא, בעצם, מה ניתן לעשות עם הנתונים המצטברים תוך כדי הפעילות השוטפת. טכנולוגיית כריית הנתונים היא אחת הטכנולוגיות העיקריות המאפשרות לארגון לעשות שימוש טוב יותר בנתונים, ולנצל משאב חשוב זה.

בפרק זה נסביר מהי טכנולוגיה זו, מדוע היא כה חשובה ומושכת תשומת לב וכיצד היא משתלבת בתפיסת מחסן הנתונים. נושא כריית הנתונים הוא נושא רחב מאוד, בחלק מהמקרים הוא יכול להפוך גם לטכני מאוד, ובקלות ניתן להקדיש לו ספר שלם. מטרת פרק זה היא לסקור בתמציתיות נושא חשוב זה, מבלי להעמיק יתר על המידה באלגוריתמים ובטכניקות המתמטיות העומדות בבסיס התפיסה.

מהי כריית נתונים

כריית הנתונים היא אחד היישומים החשובים ביותר במסגרת תפיסת מחסני הנתונים, מכיון שהיא מאפשרת זיהוי תבניות וחוקיות בנתונים. תהליך כריית הנתונים קיבל את שמו מההקבלה שניתן למצוא בין תהליך הכרייה במכרה יהלומים לאיתור מעט אבני חן עתירות ערך מבין טונות רבות של אדמה, לבין תהליך הכרייה בהר נתונים לאיתור המידע החשוב והבלתי ידוע, כלומר "היהלומים העסקיים".

קיימות הגדרות שונות למושג "כריית נתונים", כמעט כמספר החוקרים העוסקים בתחום זה וכמספר היצרנים המפתחים כלי כריית נתונים. יצרני כלי התוכנה מאמצים מושג זה

בחופשיות רבה, ולכן המציאות היא שכמעט כל יצרן מגדיר מחדש את המושג, על פי הבנתו ורצונו. אפשר כמובן להסתפק בהגדרה רחבה מאוד, הטוענת שכריית נתונים היא תהליך המבוסס על כלים שונים המאפשרים, בסופו של דבר, הבנה טובה יותר של הנתונים. תחת הגדרה רחבה זו ניתן גם לקרוא לכל מחולל שאילתות פשוט **כלי לכריית נתונים**. זה אינו המצב, וקיימת הגדרה ממוקדת יותר לכריית נתונים. מאחר ואין הגדרה אחת מוסכמת, נציג מספר הגדרות.

1: כריית נתונים היא תפיסה העוסקת בהפעלת טכנולוגיות מחשוביות ואלגוריתמים לומדים במטרה לגלות ולמצות תבניות, מגמות, סתירות וידע מתוך הנתונים.

2: כריית נתונים היא יישום של טכניקות אינטליגנציה מלאכותית (כגון Neural Networks, Fuzzy Logic, Genetic Algorithms וכדומה) במאגרי נתונים גדולים במטרה לגלות מגמות נסתרות, תבניות וקשרים בין הנתונים, וכדי להפוך ידע זה להחלטות עסקיות וביצועיות.

3: כריית נתונים היא תפיסה העוסקת באוסף של טכניקות אוטומטיות ממוחשבות שמטרתה לחקור, לגלות ולהציף יחסים וקשרים מורכבים במאגרי נתונים גדולים.

4: כריית נתונים הוא תהליך של חיפוש אחר יחסים ותבניות כלליות הקיימות בבסיסי נתונים גדולים, אך נסתרים בים הנתונים. יחסים אלה מייצגים ידע בעל משמעות לארגון ולתהליכי קבלת ההחלטות.

מקריאת הגדרות אלו ניתן לזהות את המכנה המשותף העומד בבסיס כריית הנתונים:

❖ זו תפיסה המתבססת על מיגוון רחב של טכניקות ואלגוריתמים מתוחכמים שונים, שרובם לקוחים מתחום האינטליגנציה המלאכותית והסטטיסטיקה. אין שיטה אחת המתאימה לכל הצרכים, ולכן יש להפעיל טכניקות שונות עבור בעיות שונות. כפי שנראה בהמשך, עצם ההבנה איזו טכניקה יש להפעיל עבור איזו בעיה, היא בעצמה מהווה אתגר לא טרוויאלי.

❖ זו טכניקה המבוססת על עבודה מול אוספי נתונים גדולים. כלומר, טכניקות הגילוי דורשות מספר רב של אירועים כדי שניתן יהיה לגלות תבנית או מגמה כלשהי. התבססות על מספר קטן של אירועים יכול להביא לתוצאות מוטעות.

❖ זו טכניקה המהווה חלק מתפיסת מחסן הנתונים, ואינה מחליפה אותו. כדי להפעיל את האלגוריתמים השונים צריך נתונים, ומקורם הטבעי הוא מחסן הנתונים. הסיבה לכך היא שנתונים אלה המנוהלים במחסן נתונים עברו תהליכי ניקוי, המרה, אינטגרציה ובקרת איכות. אספקת נתונים לא איכותיים לאלגוריתמים האוטומטיים יכולה להביא למסקנות חסרות כל שחר ואחיזה, וכתוצאה מכך לקבלת החלטות שגויות.

כדי להבין טוב יותר את האבחנה בין מחוללי שאילתות וכלי ניתוח סטנדרטיים, לבין טכניקות כריית הנתונים, נבחן את הנחת העבודה העומדת בבסיסה של כל אחת מהשיטות. בעיקרון, ניתן לזהות שתי קטגוריות עיקריות של תהליכי ניתוח נתונים - תהליכי ניתוח מבוססי **אימות** ותהליכי ניתוח מבוססי **גילוי**. למרות שנושאים אלה הוצגו בפרק 3 בספר, נחזור כאן על חלק מהדברים, לטובת שטף הקריאה.

❖ **תהליכי ניתוח מבוססי אימות** (Verification Based Analysis): מספר רב של כלי ניתוח מידע מבוסס על העיקרון שהמשתמש בונה הנחה כלשהי - **היפותזה** - ומתחיל להפעיל טכניקות גילוי שונות שאמורות לאושש, או לסתור, הנחה זו. תהליכים אלה מבוססים על המשתמש, על יכולתו להניח את ההנחות הנכונות, על שיקוליו בבואו לבחון את ההנחות ועל הפרשנות שלו לממצאים. במילים אחרות, תפיסה זו מבוססת לא מעט על כישוריו של המשתמש, על האינטואיציה שלו ועל יכולתו להניח הנחות ולבנות את השאלות הנכונות, כדי לבחון אותן. יעילות שיטה זו תלויה במידה רבה במספר רב של גורמים. ביניהם, ביכולת מנתח המידע לחרוג "מהקופסה", כלומר לבחון גם תבניות ומגמות לא גלויות. מחוללי השאילתות והדוחות שייכים לקטגוריה זו, ומאפשרים למשתמש לחולל בקלות משפטי SQL מורכבים, ולקבל במהירות יחסית את התוצאות. כלים אלה אינם מנחים באופן כלשהו את המשתמש איזה שאילתות עליו לשאול, ומה המסקנות המסתתרות בתוצאה המתקבלת.

כלי הניתוח הרב-מימדיים מאפשרים את הצגת הנתונים בצורה מתוחכמת ומתבססים, במידה רבה, על חישובים שונים הנעשים מראש. כלי ניתוח סטטיסטיים מאפשרים את הניתוח בין מספר משתנים, ומאפשרים להגדיר את המובהקות הסטטיסטית ביניהם. כדי להמחיש את תהליך הניתוח המבוסס על אימות נשתמש בדוגמה מתחום השיווק והדיוור הישיר. חברה מבקשת לצאת במבצע שיווק למוצר חדש, ועליה להחליט למי מלקוחותיה לשלוח מידע על המוצר חדש. מטרת הדיוור היא להגיע ללקוחות לגביהם קיימת הסבירות הגבוהה ביותר שיהיו מעוניינים לרכוש את המוצר. מנתח המידע בונה לעצמו מודל כלשהו המניח מה הן התכונות הרלוונטיות ביותר של הלקוחות (כגון אזור מגורים, רמת הכנסה, תבנית רכישות קודמות של הלקוח וכדומה). בהתבסס על מודל זה הוא מתחיל להפעיל סידרת שאילתות על הנתונים, או אמצעי ניתוח אחרים. הוא בוחן את התוצאות, מתקן את השאילתות וחוזר חלילה, עד שהוא מגיע לרשימת הלקוחות הרצויה. לא ניתן לומר שבתהליך זה נוצר ידע חדש, אך מתבצע אימות של הנחות מול הנתונים.

❖ **תהליכי ניתוח מבוססי גילוי** (Discovery Based Analysis): טכניקת כריית הנתונים המבוססת על טכניקות גילוי, פועלת באופן שונה. כאן, המשתמש אינו מניח הנחות כלשהן, אלא מאפשר לכלים האוטומטיים לפעול מול הנתונים ולהציף את התבניות המסתתרות בנתונים. אם להשתמש באנלוגיה בה כבר השתמשנו, המשתמש מאפשר לכלים האוטומטיים לגלות את היהלומים העסקיים המסתתרים בנתונים. כלי כריית הנתונים בוחנים מספר רב של משתנים בו-זמנית, ויכולים להציף יחסים שונים בין הנתונים, להצביע על קשרים מובהקים או על חריגים מיוחדים.

כריית הנתונים יוצאת מתוך הנתונים ומנסה לזהות בהם את היחסים, ההקבצות, התבניות וההקשרים. בדרך כלל פועלים תהליכים אלה באופן טוב יותר מול בסיסי נתונים גדולים, המייצגים מספר רב של תצפיות או עובדות. אם נשתמש בדוגמה הקודמת, של הגדרת אוכלוסיית הלקוחות למבצע המכירות, הרי שבטכניקות כריית הנתונים נפעיל תהליכים שונים לסיווג הלקוחות לקבוצות שונות, מבלי להניח מראש מהי קבוצת הלקוחות המתאימה ביותר למבצע. לאחר שנאתר את ההקבצות השונות האפשריות של הלקוחות, נחליט מהי הקבוצה המתאימה ביותר למבצע. במקרה זה, החיפוש אחר ההקבצות מבוצע ללא הנחה כלשהי מראש. למעשה, מתבצע כאן תהליך גילוי ההקבצות השונות, לפיהן ניתן לקבץ את אוכלוסיית הלקוחות.

מדיון זה ברור שטכניקות כריית הנתונים שונות מאוד מטכניקות מקובלות לניתוח נתונים. ניתן לומר, שההבדל העיקרי ביניהן הוא מי מניח את הנחה ובודק אותה. במקרה של תהליכים מבוססי אימות - המשתמש הוא זה המניח את ההנחה (Human generated hypothesis) ובוחן אותה. לעומת זאת, בתהליכים מבוססי גילוי, המחשב מניח את ההנחה ובודק אותה (Machine generated hypothesis). התהליכים מבוססי גילוי טומנים בחובם את ההבטחה לספק למנתחי המידע כלים מתוחכמים, כדי לאפשר להם לאתר קשרים מיוחדים ותבניות חדשות (ולא מקובלות) המסתתרות במרחב הנתונים (Data Analysis Space).

מן הראוי להדגיש שיש להתייחס לשני תהליכי הניתוח כאל תהליכים משלימים, ולא תחליפיים. הכלים לכריית נתונים אינם מחליפים את מחוללי השאילתות והדוחות השונים. הם מהווים נדבך נוסף מעל כלים אלה, כדי לאפשר מיצוי נרחב יותר של הידע מבסיס הנתונים. חלק מהידע יכול להיווצר באמצעות מחוללי שאילתות. אולם, קיים רובד נוסף של ידע המסתתר בבסיס הנתונים ואינו בר גילוי באמצעות שאילתות SQL, מתוחכמות ככל שתהיינה. לדוגמה, לא נוכל לנסח שאילתה שתבנה ותציג מהם פלחי הלקוחות, או מהן מגמות הקנייה בשנתיים האחרונות. ניתן לנסות ולענות על שאלות כאלה באמצעות שאילתות רגילות, אולם זהו תהליך מייגע הדורש הבנה מעמיקה מאוד של הפרמטרים השונים. לשם כך יש להניח מספר רב של הנחות ולהתחיל בתהליך ניתוח, שברובו מבוסס על ניסוי וטעייה (Trial and Error). תהליך כזה יכול להימשך זמן רב, ואין כל ביטחון שאכן ימצה את הידע המצוי בבסיס הנתונים.

בהכללה מסוימת ניתן לומר כך: אם אתה **יודע** מה שאתה מחפש - השתמש במחוללי שאילתות ובניתוח רב-מימדי רגיל. אם **אינך יודע** - השתמש בטכניקות כריית נתונים.

תחומי המחקר שתרמו להתפתחות כריית הנתונים

נושא כריית הנתונים מבוסס על מחקר ממספר תחומי מחקר. התחומים העיקריים הם סטטיסטיקה, אינטליגנציה מלאכותית, מערכות לומדות, תהליכי לימוד מלאכותיים ומודלים ללימוד והיסק. כל תחומי המחקר והטכניקות שהתפתחו בהם, תרמו רבות להופעת טכנולוגיית כריית הנתונים.

סטטיסטיקה

לסטטיסטיקה יש יסודות מדעיים מוצקים ופורמליים. הטכניקות הסטטיסטיות תרמו רבות להתפתחות נושא כריית הנתונים, והן משמשות כבסיס בחלק גדול מהמוצרים והיישומים. הטכניקות הסטטיסטיות הנפוצות שתרמו לנושא כריית הנתונים הן ניתוחי שונות, חישובי סטיית תקן, ניתוחי אשכולות (Cluster Analysis), ניתוחי רגרסיה (Regression Analysis), ניתוחי סדרות עתיות (Time Series Analysis), מרווחי מובהקות ושונות (Discriminant and Confidence Intervals) ועוד. הבעיה העיקרית היא שהבנת ופענוח התוצאות של המודלים הסטטיסטיים השונים דורשת מיומנות מיוחדת. כלים סטטיסטיים פופולריים כגון SPSS ו-SAS משמשים כבר משך שנים רבות לאיתור תבניות ויחסים בין נתונים מתוך מדגם גדול של נתונים. כאמור, נושא כריית הנתונים עושה שימוש נרחב בטכניקות סטטיסטיות שונות.

אינטליגנציה מלאכותית

להבדיל מהטכניקות הסטטיסטיות, האינטליגנציה המלאכותית מבוססת על ניסיון לחקות את מוח האדם, לצורך פתרון בעיות שונות. טכניקות אלה נקראות היורסטיות (Heuristics), והן דורשות בדרך כלל עוצמת חישוב גדולה בהרבה מטכניקות סטטיסטיות רגילות. רמת ההצלחה של יישומים אלה, לעומת הציפיות, היתה מוגבלת ועם מספר יישומים מצומצם, אבל המחקר בשטח זה הוליד מספר רעיונות ששימשו בסיס לנושאי מחקר אחרים.

מערכות לומדות (Machine Learning)

המערכות הלומדות מהוות הכלאה מסוימת בין טכניקות שמקורן באינטליגנציה מלאכותית וטכניקות סטטיסטיות. אחת התכונות העיקריות של מערכות המבוססות על אינטליגנציה מלאכותית היא יכולת הלמידה שלהן. כלומר, היכולת של מערכות אלו להשתנות באופן דינמי תוך כדי פעולה ולהשתפר עם הזמן, ככל שהן מטפלות ביותר ויותר מקרים. טכניקות האינטליגנציה המלאכותית מבוססות על למידה מדוגמאות, למידה באמצעות מדריך או מומחה ועוד. תהליך זה משתדל להגיע להכללה (Generalization) מהמקרים או מהדוגמאות שהמערכת פוגשת. נושא האינטליגנציה המלאכותית הוא רחב מאוד, ויכול לשמש למטרות שונות לגמרי מאשר לכריית נתונים (למשל, כדי לאמן רובוט לבצע מלאכות מסוימות). לכן, מקובל לומר שנושא כריית נתונים והפקת מידע מנתונים הוא יישום מסוים של האינטליגנציה המלאכותית. בעוד שנושא כריית הנתונים עוסק בעבודה עם מדגמים גדולים מאוד, מערכות מבוססות אינטליגנציה מלאכותית בדרך כלל עובדות עם מדגמים קטנים יחסית.

לימוד והיסק

ההיסק הוא תהליך הפקת מידע מנתונים בו נבנה מודל כלשהו על ידי ניתוח הסביבה או, במקרה שלנו, בסיס הנתונים. בדרך כלל עוסק תהליך זה בסיווג אובייקטים דומים למחלקות, ובהגדרת ובניית כללי הסיווג. כללי סיווג אלה יאפשרו בהמשך את סיווגם של אובייקטים חדשים ולא מוכרים לאחת המחלקות, או לבניית מחלקה חדשה. תהליך זה עוסק בניתוח מאפייני האובייקטים, ומציאת מאפיינים וערכם המאפשרים את הקבצת האובייקטים למחלקות שונות. במובן זה ניתן לומר שתהליך הלימוד הוא בעצם תהליך הסקת תבניות וכללים פשוטים, מתוך מכלול רחב של אובייקטים ומאפייני האובייקטים. בעיקרון, קיימות שתי טכניקות לימוד והיסק:

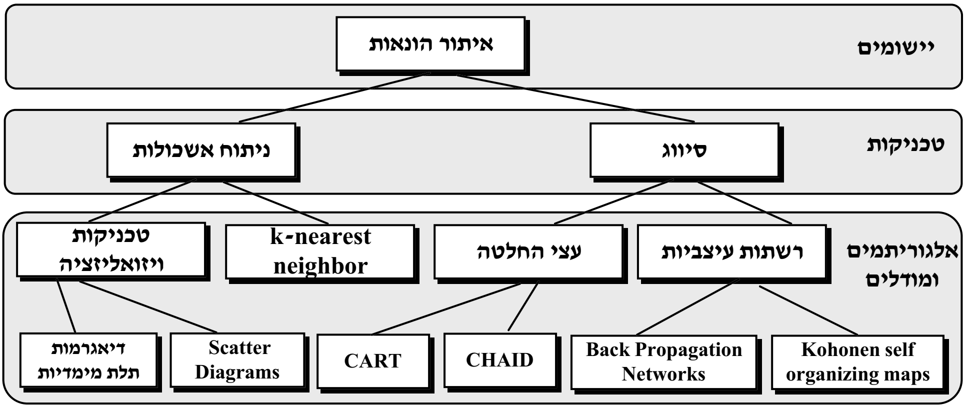
- ❖ **לימוד מונחה** (Supervised Learning) - בתהליך זה לומדים מדוגמאות. כלומר, המנחה מסייע ללומד על ידי הגדרת המחלקות ועל ידי מתן דוגמאות כיצד לשייך את האובייקטים למחלקות. המערכת הלומדת צריכה להסיק את כללי הסיווג מהדוגמאות, על ידי הבנת המאפיינים המשותפים לאובייקטים (למשל, אותו צבע, אותה צורה - אם אנו עוסקים באובייקטים גיאומטריים - או אותו גיל, אותו אזור מגורים - אם אנו עוסקים בנתונים לגבי קבוצת הלקוחות). ברגע שכללי הסיווג נבנו מהדוגמאות, ניתן להשתמש בכללים אלה כדי לסווג אובייקטים חדשים. כלומר, כאלה שלא היו חלק מהדוגמאות.
- ❖ **לימוד לא מונחה** (Unsupervised Learning) - בתהליך זה הלימוד מתבסס על תצפיות וגילוי. המערכת הלומדת מקבלת את האובייקטים, אבל שום מחלקות ודוגמאות לא ניתנות. על המערכת להסיק, ולגלות לבד מהתצפיות, מהן המחלקות האפשריות. כלומר, תהליך זיהוי המחלקות הוא תהליך לא מונחה, אלא עצמי.

הבעיה העיקרית בתהליכי הלמידה היא שקיימים מצבים אפשריים רבים ולכן, תמיד איכות הכללים שבונה המערכת אינה מבטיחה איכות סיווג מושלמת. בדרך כלל, גם לא ניתן לבחון את המודל עבור כל המצבים האפשריים.

יישומים, טכניקות, אלגוריתמים ומוצרים לכריית נתונים

דרך נוחה להתבונן ולהבין את נושא כריית הנתונים היא להתייחס לשלש שכבות שונות - יישומים, טכניקות ואלגוריתמים.

היישומים מיועדים לפתור בעיה מעשית מסוימת, כמו למשל איתור הונאות בחברת כרטיסי אשראי. ניתן לפתור את הבעיה תוך שימוש בטכניקה אחת או יותר, כמו למשל על ידי טכניקות סיווג או טכניקות ניתוח אשכולות. כל טכניקה יכולה להיתמך על ידי אלגוריתם אחד או יותר. הטכניקות והאלגוריתמים באים ארוזים במוצרים ייעודיים, או כחלק ממוצר בעל שימושים נרחבים יותר. מוצר אחד יכול לתמוך ביותר מאשר טכניקה או אלגוריתם אחד. נרחיב כעת את הדיון בשלוש השכבות האלו.



תרשים 7.1: יישומים, טכניקות ואלגוריתמים.

יישומים (Applications)

ניתן לסווג את יישומי כריית הנתונים כאוסף בעיות בעלות מאפיינים דומים, עבור נושאים שונים. הפרמטרים של היישום תלויים בכל פלח שוק או תעשיה, ולעיתים גם בין יישום ליישום. לדוגמה, ניתן להשתמש באותה טכניקה ובאותם אלגוריתמים ומודלים, כדי לבנות יישום לאיתור הונאות (Fraud Detection) עבור בנק או חברת ביטוח. ההבדל העיקרי בין יישומים אלה נובע מהפרמטרים השונים המסופקים ליישום. פרמטרים אלה שונים עבור בנק או עבור חברת ביטוח, והם מנותחים ומפורשים באופן שונה.

פרויקטי כריית נתונים נמצאים עדיין בשלב ראשוני ועדיין לא זכו לתפוצה רחבה. למרות זאת, ניתן למצוא יישומי כריית נתונים במיגוון רחב של נושאים ושווקים. מסקרים שונים עולה שהיישומים הפופולריים ביותר לכריית נתונים הם מתחום המכירות ועוסקים בפילוח לקוחות, שיווק ממוקד וניתוח סל הקניות. נסקור בקצרה מספר יישומים אופייניים.

❖ **פילוח לקוחות** (Customer Profiling) - בקטגוריה זו נמצא יישומים העוסקים בזיהוי והגדרת פרופיל הלקוחות, תוך פילוח אוסף הלקוחות לאשכולות שונים. כל זאת במטרה להבין מהן תכונותיו של לקוח "טוב ונאמן", לצורך חיזוי קהל המטרה למבצעים שיווקיים שונים. יישומים אלה יכולים למצוא תבניות תכונות מקובץ הלקוחות הנוכחי, ולהשתמש בתוצאות הפילוח כדי להגדיר את קהל לקוחות המטרה למבצעים שונים.

❖ **ניתוח נטישה** (Churn Management) - בקטגוריה זו נמצא יישומים העוסקים בניתוח הלקוחות שנטשו את החברה לטובת המתחרים, כדי להבין את הסיבות לעזיבה ואת תבנית תכונות לקוח שעזב. בדרך זו ניתן להחליט כיצד לנהוג ואילו שיטות להפעיל כדי למנוע נטישה. כידוע, שמירה על לקוח הוא תהליך זול בהרבה מאשר רכישת לקוח חדש, לכן, יישומים אלה הם בין החשובים ביותר המופעלים בארגונים.

חלק מהיישומים מאפשרים להגדיר הסתברות נטישה ולתת כיווני מחשבה על הפעולות שהחברה יכולה לנקוט בכל מצב. למשל, אם הסתברות הנטישה בחברת טלקומוניקציה היא מעל 80% ניתן להציע ללקוח קו טלפון נוסף חינם. במקרה שהסתברות הנטישה נמוכה יותר, נניח 50%, ניתן להציע ללקוח 100 דקות שיחות חינם כבונוס מיוחד. איתור מוקדם של לקוחות פוטנציאליים לנטישה וטיפול בהם יכול להקטין באופן דרמטי את מספר הלקוחות הנוטשים.

- **ניתוח סל הקניות** (Market Basket Analysis) - בקטגוריה זו נמצא יישומים העוסקים בניתוח הרגלי הקנייה של לקוחות, במטרה להבין איזה מוצרים ירכשו על ידי לקוח מסוים, לאורך זמן. הבנה זו מסייעת לארגונים לקבוע איזה מוצרים כדאי למכור לאיזה לקוחות (Cross Selling), לקבוע רמות מלאי בחנויות, לתכנן את שטח המדפים וצורת הצגת המוצרים וכדומה.

- **שיווק ממוקד** (Targeted Marketing) - בקטגוריה זו נמצא יישומים העוסקים בניתוח סל הקניות של לקוחות, במטרה להגדיר את אוכלוסיית המטרה למבצעי שיווק שונים, למבצעי דיוור ישירים (Direct Mail Campaigns), לקבוע לאיזה לקוחות כדאי לשלוח איזה קטלוגים של מוצרים, ניתוח יעילות מבצעי שיווק וכדומה. יישומים אלה מבוססים על טכניקות שונות לניתוח סל קניות של הלקוח.

- **איתור הונאות** (Fraud Detection) - בקטגוריה זו נמצא יישומים העוסקים באיתור הונאות פוטנציאליות במיגוון רחב של תחומים (כגון בנקאות, מסחר ניירות ערך, טלקומוניקציה, ביטוח, שירותי בריאות וכדומה). בין היישומים ניתן למנות איתור שימוש חריג בכרטיסי אשראי, איתור שימוש חריג בשיחות טלפון בינלאומיות וכדומה.

- **דירוג אשראי** (Credit Scoring) - בקטגוריה זו נמצא יישומים העוסקים בניתוח הסיכונים במתן אשראי ללקוחות, זיהוי לקוחות העומדים לשנות את דירוג האשראי שלהם, זיהוי יחסים לא גלויים בין אינדיקטורים פיננסיים שונים.

- **יישומי ביטוח ובריאות** (Insurance and Health Care) - בקטגוריה זו נמצא יישומים העוסקים בניתוח תביעות מבוטחים כגון, אילו טיפולים רפואיים בדרך כלל מתבצעים ביחד, זיהוי לקוחות שיירכשו פוליסות חדשות, זיהוי תבניות התנהגות של לקוחות מסוכנים, זיהוי הטיפולים המוצלחים למצבי מחלה מסוימים וכדומה.

- **בקרת איכות** (Quality Control) - בקטגוריה זו נמצא יישומים העוסקים בניתוח תקלות בקווי הייצור, תוך ניסיון לחזות גורמי תקלה, תקלות בתהליכי הייצור, ניתוח תלונות לקוחות אודות פגמים במוצר וכדומה. כל זאת במטרה לשפר את תהליכי הייצור ואת איכות המוצרים.

לסיכום סקירה קצרה זו נאמר שהפוטנציאל ביישומי כריית נתונים הוא גדול, ועם הזמן הולך וגדל עוד יותר. מכיון שרוב יישומים אלה הם יישומים אסטרטגיים עבור הארגונים, הם אינם נוטים לפרסם ברבים את השימוש בהם, ומתייחסים אליהם כאל נושאים בעלי יתרון תחרותי.

טכניקות (Techniques)

כל יישום כריית נתונים נתמך על ידי אוסף טכניקות המשמש להפקת הידע מהנתונים. הטכניקות נבדלות ביניהן בסוגי הבעיות אותן הן מסוגלות ומיועדות לפתור. בשוק קיימים כיום כלים התומכים במספר טכניקות בו-זמנית, ולעיתים אותה טכניקה נתמכת על ידי יותר מאלגוריתם אחד. נסקור בקצרה את הטכניקות העיקריות המקובלות בכריית נתונים.

הצגה ויזואלית של נתונים (Visualization Techniques)

השימוש בטכניקות פשוטות ומתוחכמות להצגת נתונים נפוץ מאוד בתהליכי כריית נתונים. עצם היכולת להציג את הנתונים למשתמש באופן ויזואלי מהווה שיפור ביכולת המשתמש להבין ולפענח את אותם נתונים. קיימות מספר רב של טכניקות ויזואליזציה העושות שימוש בצבעים, בהצגה תלת מימדית (3D Visualization), ביכולות דינמיות של התבוננות בתמונה מכיוונים שונים, וכדומה. אחת הטכניקות הנפוצות ביותר היא דיאגרמת פיזור (Scatter Diagrams). דיאגרמה דו-מימדית זו מציגה את פיזור מרחב הנתונים, לפי שני פרמטרים בכל פעם. למשל, בחברה המוכרת תקליטים ניתן להציג מה פיזור האוכלוסיה, לפי רמת הכנסה וגיל כאשר ההתייחסות היא לכל לקוחות החברה. מהתבוננות בדיאגרמת הפיזור ניתן, למשל, לגלות שרוב רוכשי התקליטים הם צעירים בעלי רמת הכנסה בינונית.



תרשים 7.2: דוגמה לדיאגרמת פיזור.

הסקת חוקים ויחסים בין הנתונים (Association Rules)

טכניקות גילוי החוקים והיחסים בין נתונים, ממוקדות בזיהוי איזה נתונים קשורים באופן הדוק עם נתון אחר. תוצאה של טכניקות אלו היא, בדרך כלל, הצגת חוקיות הסתברותית כלשהי, כדוגמת "אם מוצר A נרכש, אזי ההסתברות לרכישת מוצר B באותו אירוע היא P". טכניקות הסקת החוקים והיחסים מתאימות במיוחד עבור אוסף בעיות שמקובל לכנותן בשם **ניתוח סל הקניות** (Market Basket). בעיות אלו עוסקות בניתוח סל הקניות של הלקוח (למשל, המוצרים שלקוח קונה במרכול) במסגרת עסקת רכישה אחת בודדת. המטרה היא למצוא את המגמות, הרגלי הקנייה והיחסים בין סוגי המוצרים הנרכשים, וכל זאת בהתבסס על מדגם גדול מאוד של עסקאות. הידע הנוצר כתוצאה מתהליך ניתוח זה יכול לשמש לקביעת רמות המלאי המוצרים השונים בחנות, לקבוע את צורת ארגון המוצרים על המדפים בחנות, לקבוע אילו מוצרים כדאי לשתף במבצעי הנחות ועוד.

המקור של טכניקות אלו הוא ברשתות המזון, אך ניתן להפעיל אותן גם בפלחי שוק אחרים כמו למשל, אילו שירותים קונים לקוחות יחד בשוק הבנקאי, אילו שירותים כדאי להציע יחד בשוק הטלקומוניקציה, אילו טיפולים רפואיים כדאי לתת יחד עם טיפולים אחרים וכדומה.

בדרך כלל, מחזירות טכניקות אלו תוצאות המבוססות על אחוזי מובהקות. למשל:

- ❖ 80% מכלל הרכישות בהן נרכשה גבינה לבנה דלת שומן, כללו גם רכישת מעדן חלב דל שומן אחר כלשהו;
- ❖ 85% מהרכישות בהן נרכשה שתייה קלה כלשהי, כללו גם רכישה של חטיפים מלוחים כלשהם;
- ❖ 92% מהלקוחות להם יש מנוי לירחון העוסק בבתים וגינות ומנוי לירחון העוסק במוסיקה, יש להם גם מנוי לירחון העוסק במכוניות;
- ❖ 84% מהפעמים בהם נרכש טוסטר מתוצרת מסוימת נרכשה גם מסחטת מיץ.

רוב הכלים מאפשרים קביעת סף כלשהו של אחוזי מובהקות, כדי למנוע הצפת מקרים טרוויאליים. טכניקה זו היא איטרטיבית באופייה כלומר, הצפת יחסים מסוימים יכולה להוביל לניסיון לנתח יחסים חדשים או לנתח פרמטרים חדשים.

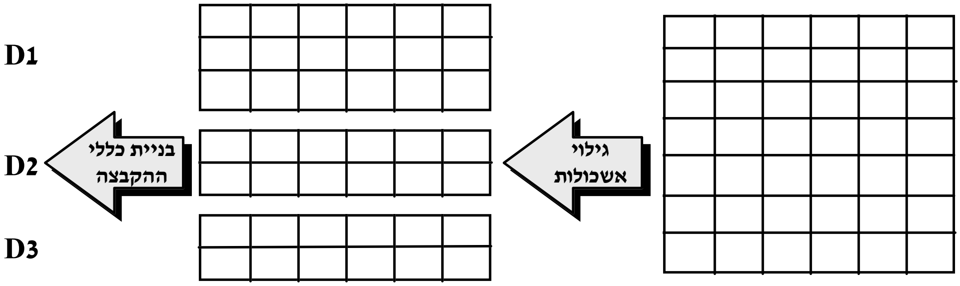
ניתוח סדר הופעה (Sequence Based Analysis)

טכניקות הניתוח והסקת החוקים והיחסים הרגילות בסל הקניות, עוסקות באוסף מוצרים הנרכשים בעסקה אחת בודדת בנקודת זמן כלשהי. שכלול מסוים של טכניקה זו היא להתבונן על אוסף עסקאות על ציר הזמן, ביניהן יש מכנה משותף כלשהו. למשל, מספר כרטיס אשראי, מספר חשבון, מספר חבר במועדון הנוסע המתמיד וכדומה. בניתוח זה, מעבר לחשיבות שבהופעת מוצרים שונים באותו סל קנייה, מנסים גם להבין את תבנית וסדר ביצוע העסקאות, כפי שהן משתקפות על ציר הזמן.

מניתוח זה יכולים לחזות תבניות רכישה מסוימות ולצפות מראש לרכישה עתידית. לדוגמה, בחברה העוסקת בביטוח רפואי ניתן לאתר תהליכי טיפול עתידיים, הצפויים כתוצאה ממחלה או תרופה מסוימת. חוקיות שמתגלה כתוצאה מהפעלת טכניקה זו יכולה להיות "אם ניתן טיפול T אזי ההסתברות להופעת דלקת Y לאחר שבועיים היא P". חברה העוסקת במסחר במניות יכולה להשתמש בטכניקה זו כדי לאתר חוקיות בתהליך המסחר, ולמצוא חוק כגון "אם מניה A עולה ביותר מ- 20%, בעוד מדד המניות יורד, אזי מניה B תעלה כעבור שבוע בהסתברות P".

ניתוח אשכולות (Cluster Analysis)

טכניקות אלו עוסקות בפילוח אוכלוסיות אובייקטים גדולות לאוסף כלשהו של אשכולות בעלי הגיון פנימי כלשהו. כלומר, קיים דמיון מסוים בין האובייקטים השייכים לאשכול, בהתבסס על קריטריון כלשהו. המטרה היא למצוא הקבצות השונות אחת מהשנייה, ואת הדמיון בין חברי ההקבצה.



תרשים 7.3: תהליך בניית האשכולות.

תהליך השיוך לאשכול כלשהו מבוצע באופן אוטומטי על ידי אלגוריתמים לבניית אשכולות המזהים את המאפיינים המשותפים לאוכלוסיה, ומבצעים את החלוקה לאשכולות. החכמה באלגוריתמים אלה היא שאין צורך לקבוע מראש אילו פרמטרים ישמשו לקביעת החלוקה, או להגדיר מראש אילו אשכולות יש לבנות. הבנת ופענוח תוצאות ההקבצה מחייבים מיומנות והבנה טובה בנתונים. בדרך כלל, משמשים אלגוריתמים אלה לפילוח אוכלוסיית הלקוחות לאשכולות של לקוחות בעלי מאפיינים דומים. תוצאות אלגוריתמים אלה משמשים כנקודת פתיחה טובה לחקירה וניתוח מעמיקים יותר.

סיווג (Classification)

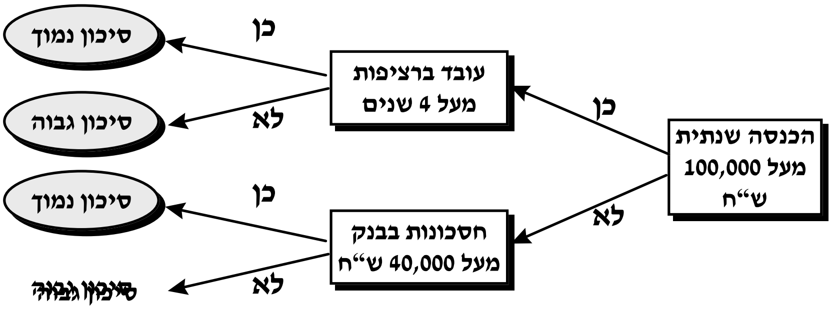
טכניקה זו היא אחת הנפוצות בתחום כריית הנתונים, ומשמשת בעיקר לתהליכי חיזוי. טכניקת הסיווג מאפשרת לבנות מודל סיווג כלשהו מבסיס נתונים היסטורי. בדרך כלל, טכניקת הסיווג משתמשת בהקבצות ידועות מהנתונים כדי לפתח את שיטת הסיווג, או להגדרת הקבצות חדשות ובלתי ידועות מהנתונים. שימושים כגון קביעת קהל מטרה למבצעי שיווק ממוקדים, איתור הונאות וניתוח סיכונים במתן אשראי, מתאימים

במיוחד לטכניקה זו. גישה זו מבוססת, בדרך כלל, על אלגוריתמים עצי החלטות, או רשתות עצביות. קיים קשר בין טכניקות חיזוי וטכניקות סיווג - ניסיון לחזות אם לקוח מסוים ינהג בדרך מסוימת, דומה לבניית הנחה שהלקוח משתייך לקבוצת סיווג מסוימת. נציג בפירוט נוסף את השימוש בשתי טכניקות עיקריות אלו לפתרון בעיות סיווג.

עצי החלטה (Decision Trees)

עצי החלטה הם טכניקה המאפשרת זיהוי גורמים משפיעים, ולמיין אותם על פי סדר החשיבות שלהם. צורת הצגת תוצאות הניתוח היא של עץ היוצא משורש, מסתעף למספר ענפים ומסתיים בעלים. עצי החלטה הם דרך נוחה לייצוג ידע, מכיון שהם מאפשרים הקבצת אוכלוסיה מסוימת לאוסף הקבצות סופי. האובייקטים מסווגים על ידי מעבר בעץ ההחלטות, מהשורש שלו ועד לעלה בקצה העץ, וכל זאת על סמך הערכים השונים הנמצאים בצמתי הביניים.

תהליך השימוש בעצי החלטה מבוסס על החיפוש איזו תכונה מנבאת באופן הטוב ביותר התנהגות מסוימת. נתחיל בדוגמה המציגה עץ החלטות להגדרת לקוח מבחינת רמת הסיכון במתן האשראי. מניתוח אוכלוסיית לקוחות הבנק שקיבלו אשראי והחזירו אותו, או שהיו בעיות בהחזרת האשראי, ייתכן שנקבל את עץ ההחלטות הבא:

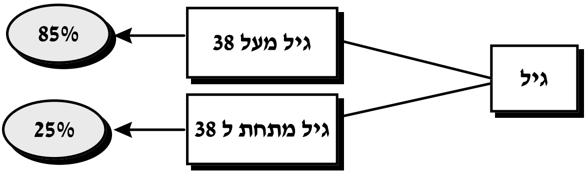


תרשים 7.4: עץ החלטות לקביעת סיכוני אשראי.

עץ ההחלטות המוצג כאן מאפשר למי שצריך לקבל החלטה לגבי מתן אשראי לברר מספר נתונים אודות הלקוח, ועל סמך כך להבין את רמת הסיכון במתן האשראי. מובן שהעץ המוצג כאן הוא פשטני מאוד. במציאות, בניית עץ החלטות להגדרת סיכוני אשראי הוא תהליך מורכב הרבה יותר. בסופו של תהליך עשוי להתקבל עץ עם מספר רב מאוד של ענפים והסתעפויות.

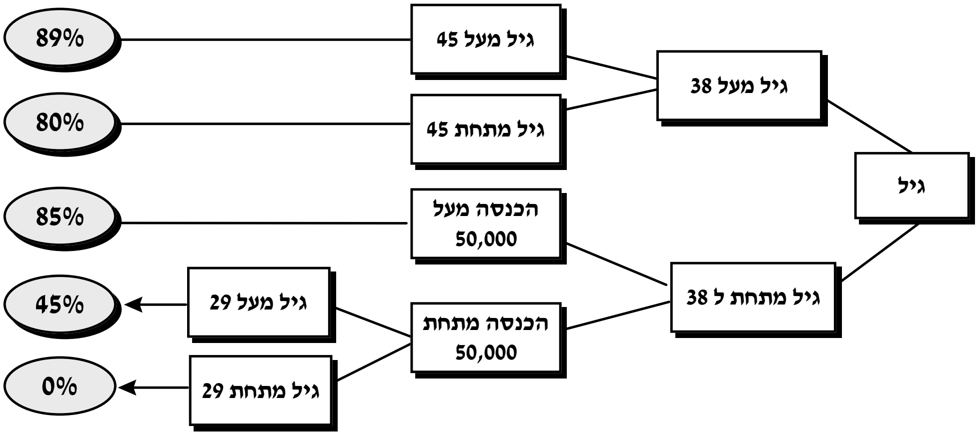
דוגמה אחרת לשימוש בעצי החלטות תהיה לחיזוי הפיכת **לקוח בפוטנציה** ל**לקוח בפועל**. אם ננתח את אוכלוסיית היעד של הלקוחות הפוטנציאליים (Prospects) ואנו יודעים את גילו של כל לקוח, רמת הכנסתו ואזור מגוריו, נשאלת השאלה אילו מהתכונות המוזכרות כאן מנבאת באופן הטוב ביותר את הפיכתו מ**לקוח פוטנציאלי** ל**לקוח אמיתי** (Customers). אם אנו משערים שגיל הלקוח הוא המנבא הטוב ביותר, עלינו למצוא אם קיים סף מסוים המבדיל בין לקוחות פוטנציאליים ללקוחות אמיתיים.

קיימים אלגוריתמים להסקת עצי החלטות הסורקים את בסיס הנתונים, ועל סמך בחינת גיל הרוכשים לעומת כלל האוכלוסיה יכולים למצוא ערכי סף. בתרשים הבא ניתן לראות עץ החלטה המציג מצב בו גיל הסף הוא 38. כל מי שגילו מעל 38, ההסתברות להפיכתו לרוכש הוא 85%, לעומת הסתברות של 25% אם גילו מתחת ל-38.



תרשים 7.5: עץ החלטות עבור הגיל.

אם נמשיך לחפש אחר עץ ההחלטות המנבא בצורה הטובה ביותר את הפיכת הלקוח הפוטנציאלי ללקוח בפועל, נוכל לקבל עץ החלטות כמוצג בתרשים הבא.



תרשים 7.6: עץ החלטות עבור מספר תכונות.

מתרשים זה ניתן לומר שלקוח שהוא מתחת לגיל 29, ורמת הכנסתו מתחת ל-50,000 ש״ח, לא יהפוך ללקוח בפועל. לעומת זאת, לקוח שגילו מעל 45 יהפוך ללקוח בפועל, כך בהסתברות של 89%.

בנוסף להצגה הוויזואלית של עץ ההחלטות, ניתן להפוך את העץ לאוסף כללי החלטה, כלל אחד עבור כל עלה בעץ. לדוגמה:

- אם "**גיל מעל 45**" אז סיכוי הפיכתו ללקוח הוא **89%**
- אם "**גיל בין 38 ל-45**" אז סיכוי הפיכתו ללקוח הוא **80%**
- אם "**גיל מתחת 38**" וגם "**הכנסה מעל 50,000**" אז סיכוי הפיכתו ללקוח הוא **85%**
- אם "**גיל בין 29 ל-38**" וגם "**הכנסה מתחת ל-50,000**" אז סיכוי הפיכתו ללקוח הוא **45%**
- אם "**גיל מתחת 29**" וגם "**הכנסה מתחת 50,000**" אז סיכוי הפיכתו ללקוח הוא **0%**

השימוש בעצי החלטות מתבצע בדרך כלל במספר שלבים:

- ❖ **בניית העץ** (Decision Tree Induction) - בשלב זה מציגים לאלגוריתם את הנתונים ונותנים לו לבנות את העץ הנובע מהם.
- ❖ **בדיקת סבירות ופישוט העץ** (Tree Prunning) - בשלב זה בודקים את סבירות העץ שהתקבל ומבצעים בו מספר תיקונים, למשל על ידי איחוד ענפים, פיצול ענפים, שינוי תנאי סף וכדומה. שלב זה מתבצע על ידי מומחי יישום, המכירים את נתוני האימון ומסוגלים לבדוק את איכות התוצאות. בדרך כלל, מקובל בשלב זה להפעיל את האלגוריתם על אוסף נתונים נוסף ולבדוק את סבירות התוצאות המתקבלות.
- ❖ **קבלת החלטות** - בשלב זה משתמשים בעץ לקבלת החלטות. בדוגמה שלנו, ניתן להציג לאלגוריתם נתוני לקוחות פוטנציאליים ולתת לו לסווג אותם. בתוצאות ניתן להשתמש לצורך מבצע שיווק ישיר על ידי פנייה אל אותם אנשים. יישום אחר של עצי החלטה יכול להיות לצורך חיזוי סיכוני אשראי (Credit Risk Predicting). לאחר שמציגים לאלגוריתם אוסף נתוני לקוחות לא מסוכנים, וכאלה שלא עמדו בהחזר האשראי - בונים את עץ ההחלטות. כעת, ניתן להפעילו בנפרד לגבי כל לקוח של הבנק המבקש לקבל אשראי.

עצי החלטות הפכו לכלי כריית נתונים פופולרי מאוד בשל היותם קלים להבנה ובעלי רמת דיוק גבוהה למדי. המשתמשים יכולים להבין בקלות את התוצאות, להבדיל משיטות אחרות בהן התוצאה מוצגת כ"קופסה שחורה" ללא יכולת להסביר כיצד האלגוריתם הגיע אליה. בדרך כלל, תהליך בניית עצי ההחלטות קצר יותר ממשך הזמן שלוקח להשתמש בטכניקות אחרות, כגון רשתות עצביות. צורת ההצגה הוויזואלית של עצי ההחלטות, והיכולת להפיכתם בקלות סבירה לאוסף כללים ברורים, הפכה מודל זה לנוח וקל מאוד להבנה.

יחד עם יתרונות אלה, צריך לציין שלטכניקת עצי החלטות יש גם מספר חסרונות - עצים אלה אינם מתאימים לכל סוג נתונים, לעיתים התוצאות המתקבלות מורכבות ודורשות תהליכי פישוט. ברוב המוצרים הנפוצים לכריית נתונים מופיעים מספר אלגוריתמים לבניית עצי החלטות. בנוסף לכלים ייעודיים לכריית נתונים, כגון Intelligent Miner של חברת IBM או MineSet של חברת Silicon Graphics, הוסיפו גם חלק מיצרני כלי OLAP מודול נוסף לבניית עצי החלטות, כדי לאפשר למשתמשים צורת ניתוח הנתונים נוספת. לדוגמה, ב- Business Objects קיים מודול המאפשר בניית עצי החלטות מהנתונים, וזאת בנוסף למודול מחולל שאילתות ומודול ניתוח רב-מימדי.

טכניקת עצי ההחלטה התפתחה כתוצאה ממחקר שבוצע באוניברסיטת מישיגן, ונקרא AID - Automatic Interaction Detection. שיטה זו מבוססת על אלגוריתם הבודק באופן אוטומטי את הערכים, ומזהה את אותם ערכים הקשורים באופן חזק לערכי פלט אותם בודקים. הערכים אותם מוצא האלגוריתם הם בעלי יכולת להסביר ולנבא את הנתונים. בהמשך פותח אלגוריתם נוסף, שנקרא CHAID - Chi Squared AID שהרחיב את יכולות האלגוריתם הקודם, על ידי הוספת הטכניקה הסטטיסטית, Chi בריבוע.

חוקר אוסטרלי, בשם Ross Quanilan מאוניברסיטת סידני, נחשב לאבי נושא עצי החלטות לאחר שפיתח בשנת 1983 אלגוריתם לו קרא ID3. אלגוריתם זה מפתח עצי החלטות ומסוגל למיין אותם על פי סדר החשיבות. בהמשך התפתחו אלגוריתמים מתקדמים יותר כגון ID4, ID6, C4.5 ו-Gini.

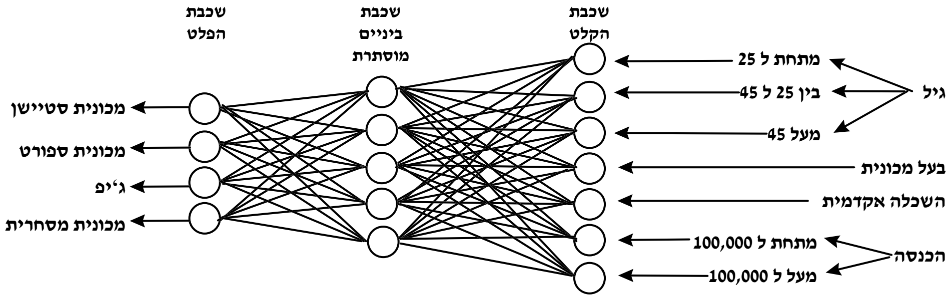
רשתות עצביות (Neural Networks)

ניתן לפתור בעיות סיווג גם באמצעות ענף חדש יחסית של אינטליגנציה מלאכותית, רשתות עצביות. זהו ענף של מדעי המחשב המבוסס על פיתוח מבנים מתמטיים בעלי יכולת לימוד עצמית. הבסיס העומד מאחורי ענף מחקרי זה נולד משילוב ענפי מחקר העוסקים בחקר המוח ובחקר תהליכי למידה. מקובל לתאר את המוח כרשת עצבית המורכבת ממספר גדול מאוד של נוירונים (Neurons) המחוברים אחד עם השני באמצעות סינפסות (Synapses), ולכן מתקבלת רשת עצבית ענקית. נוירון בודד יכול להיות מחובר לנוירונים אחרים באמצעות אלפי סינפסות. מנגנון פשוט זה מאפשר לבצע פעולות מורכבות ביותר, ומשמש כבסיס לאחד המנגנונים המופלאים ביותר - מנגנון הלמידה.

מבנה זה שימש חוקרים שונים לבניית מודלים ומכונות בעלי יכולת למידה. המודל מבוסס על אוסף צמתים (המחולקים לצמתי קלט, צמתי ביניים וצמתי פלט) המחוברים ביניהם ברשת גדולה. **צמתי הקלט** מקבלים את הבעיה ו**צמתי הפלט** מציגים את התוצאה. **צמתי הביניים** משתתפים בתהליך פתרון הבעיה. השימוש ברשתות עצביות מורכב משני שלבים עיקריים - שלב הקידוד (Encoding), בו מציגים לרשת אוסף דוגמאות ומאמנים את המערכת לפתור בעיות מסוימות, ושלב הפענוח (Decoding), בו מציגים למערכת מקרים ומשתמשים בהם לביצוע פעולות סיווג, חיזוי ולימוד נוסף. במובן מסוים, ניתן להתייחס לרשת כזו, לאחר שעברה את תהליך האימון, כאל "מומחה" בתחום מסוים.

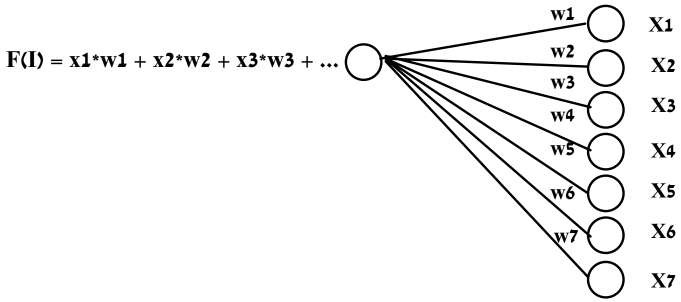
כבר לקראת סוף שנות ה-50, בנה חוקר מאוניברסיטת Cornell שבמדינת ניו-יורק, מערכת לה קרא בשם Perceptron. ככל הידוע, מערכת זו היא היישום הראשון של רעיון הרשתות העצביות. במהרה, הוכיחו מדענים שבעיות אותן ניתן לפתור באמצעות ארכיטקטורה זו, מוגבלות למדי. רק בשנות ה-80 החלו להתפתח ארכיטקטורות מתוחכמות יותר, המבוססות על שכבות ביניים ועל שיטת Back Propagation. הרעיון הוא, שכל הצמתים בשכבת הביניים מכילים בתחילה משקלות אקראיות. בשלב הלימוד מציגים לרשת דוגמאות ומשנים ומכוונים את המשקלות, עד לקבלת התוצאה הרצויה. לאחר אימון סביר של הרשת, היא יכולה להפוך לכלי חיזוי טוב מאוד.

התרשים הבא מציג רשת עצבית, שמטרתה לסווג איזה לקוחות יקנו איזה סוג של מכוניות. הסיווג מתבסס על ארבעה פרמטרים - גילם, האם כיום יש ברשותם מכונית, האם הם בעלי השכלה אקדמית ורמת הכנסתם. שים לב, שהפרמטר **גיל** מחולק לשלושה טווחים ואילו הפרמטר **רמת הכנסה** מחולק לשני טווחים. מובן, שבמציאות מספר הפרמטרים גדול בהרבה, ולכן בעיית הסיווג היא בעיה מורכבת.



תרשים 7.7: רשת עצבית לסיווג רוכשי מכוניות.

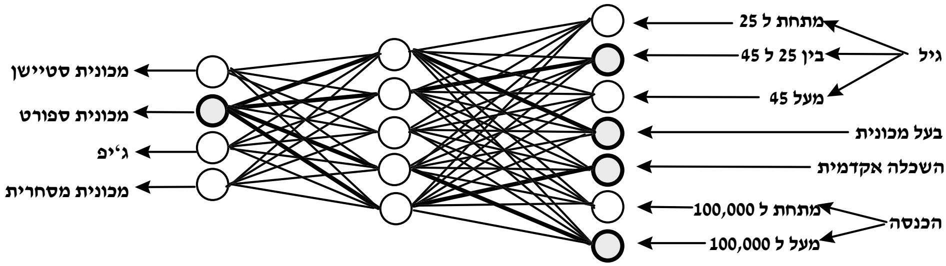
כפי שניתן לראות מהתרשים, כל צומת בשכבת הביניים מקושרת לכל צמתי הקלט ולכל צמתי הפלט. משמעות הדבר היא שכל מה שהרשת לומדת מבוסס על כל הקלטים המתקבלים. לכל ערוץ קלט המגיע לצומת ביניים נקבע משקל מסוים, והצומת מבצעת חישוב המבוסס על שקלול כלשהו של כל הקלטים. משקל כל צומת נקבע תוך כדי תהליך אימון הרשת, בו מציגים לה את הקלטים השונים, יחד עם הפלט הרצוי.



תרשים 7.8: שקלול הקלטים בצומת בשכבת הביניים.

מובן שזו הצגה פשטנית של הרשת העצבית, ובמציאות האלגוריתמים לשקלול מורכבים יותר. מקובל לבצע טרנספורמציות לא לינאריות על תוצאות השקלול, לשיפור תוצאות החיזוי של הרשת.

בנוסף לקביעת משקל הערוצים השונים בכל צומת בשכבת הביניים, יש להגדיר את הקלטים עצמם. ההחלטה לגבי הפרמטרים והטווחים אינה החלטה טרוויאלית. היא דורשת עבודה מקדימה המבוססת, בין היתר, גם על הנתונים הזמינים אודות כל לקוח. אם למשל, אין בידי החברה נתונים לגבי התפלגות ההכנסות, אזי אין כל הגיון להגדיר פרמטר זה. לחילופין, יש לעשות מאמץ להשיג נתונים אלה באופן ישיר, על ידי פנייה ללקוחות, או באופן עקיף - למשל דרך אזור מגורי הלקוח. על סמך נתוני מכירות העבר, מתחילים את שלב הקידוד, בו מציגים לרשת דוגמאות של לקוחות שקנו מכוניות, ואת הפרמטרים שלהם, ומשחקים עם המשקלות עד לקבלת התוצאה הנכונה. תרשים 7.9 מציג דוגמה אחת של פרמטרים של לקוח - גיל 32, בעל מכונית, בעל השכלה אקדמית ובעל הכנסה מעל 100,000 ש"ח בשנה - שרכש מהחברה מכונית ספורט.



תרשים 7.9: רשת עצבית במצב קידוד.

בדרך זו מתחילים להציג למערכת אוסף גדול של דוגמאות, ומשחקים עם המשקלות כך שנקבל את התוצאה הרצויה. תהליך זה יכול לקחת זמן רב, ודורש מיומנות והבנת הפרמטרים והמשקלות. בגמר התהליך, ניתן להתחיל להציג למערכת אוכלוסיית לקוחות פוטנציאליים גדולה, ולהשתמש בה לביצוע הסיווג. בהתבסס על תוצאות הסיווג, ניתן לשלוח בדיוור ישיר פרסומים ממוקדים לגבי היצע הדגמים השונים הרלוונטיים לסוג הלקוח.

דוגמה נוספת לשימוש ברשתות עצביות יכולה להיות עבור יישום לאיתור הונאות בתחום הטלקומוניקציה. בשלב הקידוד, ייבנה אוסף רשומות שיחות טלפון, המייצגות מקרי הונאה (למשל, שיחות ארוכות מאוד ליעדים מסוימים, מספר שיחות רב ליעדים מסוימים בין שעות מסוימות וכדומה), וכן דוגמאות לשימוש תקין. בשלב ראשון, יוגדרו מאוכלוסיות הדוגמה מהם הפרמטרים הרלוונטיים, כדי לאתר את מצב ההונאה ואת מאפייניו. עם בניית הפרמטרים, המאפיינים ומתן השקלול לצמתים (על סמך אוכלוסיית הדוגמה), מופעל האלגוריתם על האוכלוסיה האמיתית. כעת, הוא ינסה לאתר את המצבים הדומים. שימוש נוסף יכול להיות הפעלת טכניקה לסיווג לקוחות שמסוכן לתת להם אשראי.

השימושים המתאימים ביותר לרשתות עצביות הם עבור מודלים לא לינארים. כלומר, לא ניתן לייצג את הנתונים על ידי משוואות לינאריות, או עבור בסיסי נתונים המכילים הרבה נתונים לא אמינים, או נתונים חסרים. כמו כן, אלגוריתמים אלה עובדים רק על נתונים נומריים, דבר הגורם לקשיים בייצוג הנתונים. לדוגמה, אם אחד הפרמטרים הוא **אזור גיאוגרפי** וקיימים 20 אזורים גיאוגרפיים, יהיה צורך להכין רשומת קלט שתכיל 20 שדות המכילים אפס, או אחד, בהתאם. משמעות הדבר, יש צורך בתהליך הכנת הנתונים במבנה מיוחד עבור האלגוריתם, ובדרך כלל לא ניתן להפעילו באופן ישיר על הנתונים כפי שהם מאוחסנים במחסן הנתונים.

אחת הבעיות העיקריות בשימוש ברשתות עצביות היא שתוצאות הסיווג המתקבלות קשות להסבר, ולכן מקובל להתייחס לרשתות אלה כאל "קופסה שחורה". העדר יכולת ההסבר מונעת את הרחבת השימוש בטכניקה זו, למרות התוצאות הטובות המתקבלות. חסרון נוסף הוא שתהליך אימון רשת יכול להיות ארוך, וקיים קושי להגדיר מתי להפסיק את האימון. למרות חסרונות אלה, השיפורים המתמידים בטכניקה זו הפכו אותה לאחת הטכניקות הנפוצות לחיזוי.

אומדן (Estimation)

זו טכניקה מסוימת, המהווה שיפור מסוים לטכניקות הסיווג הרגילות, שבדרך כלל נותנות תוצאה בינארית - כן או לא. בטכניקות האומדן מפתחים אוסף ציונים, לא תשובה בינארית. לדוגמה, במקרה של ניתוח סיכוני אשראי, מפתח האלגוריתם ציון מסוים המצביע על כדאיות והסיכון במתן האשראי.

טכניקות אחרות

בנוסף לגישות שהוזכרו, התפתחו גישות נוספות בהן ניתן להשתמש בנפרד, או במשולב עם טכניקה אחרת כלשהי. בין גישות אלו ניתן לציין גישות כמו Case Based Reasoning, לוגיקה עמומה (Fuzzy Logic), אלגוריתמים גנטיים (Genetic Algorithms), המרות פרקטליות (Fractal Based Transforms) ועוד.

טבלת סיכום יתרונות וחסרונות הטכניקות השונות

כפי שהסקירה הקצרה שהובאה לעיל מראה, לכל אחת מהטכניקות יש יתרונות וחסרונות, ולכן הן מתאימות לסוג מסוים של יישומים בלבד. ננסה לתמצת את היתרונות והחסרונות של טכניקות אלה בטבלה אחת:

טכניקה	יתרונות	חסרונות
עצי החלטות	מספקת הבנה טובה של המודל והנתונים. צורת הצגת התוצאות נוחה להבנה על ידי רוב המשתמשים. שילוב הצגת חוקים יחד עם הצגה גרפית מקלה מאוד את השימוש.	במצבים מסוימים יכולים להתקבל עצים מורכבים מאוד שהופכים לבלתי מובנים וקשים מאוד לניתוח.
רשתות עצביות	מתאים לבניית מודלים סטטיסטיים המבוססים על נפחי נתונים גדולים, הכוללים גם נתונים מזוהמים.	"קופסה שחורה" - למרות הדיוק בהצגת המודל, קיים קושי בהסבר התוצאות, דבר המרתיע את המשתמשים.
כלי ויזואליזציה	מאפשרים הצגה ויזואלית דו/תלת מימדית, תוך שילוב אנימציה. מאפשרים למשתמשים לגלות תבניות המסתתרות בנתונים.	תלויים באופן מוחלט ביכולת המשתמש להבין ולגלות את תבניות המידע.
כלים סטטיסטיים	מספר רב של אלגוריתמים שונים, מאפשר למשתמשים מנוסים להסיק מסקנות מהנתונים.	מתאים יותר לניתוח נתונים מאשר להסקת מידע בלתי ידוע ותבניות חדשות מהנתונים.
מערכות מומחה	מתאימות לבעיות מסוימות ויכולות להביא לתוצאות טובות מאוד ביישומים מסוימים.	מבוססות על טיב הידע המסופק על ידי המומחה האנושי, וביכולות הכלי לנסח את הידע המוצג בפניו.

אלגוריתמים ומודלים (Algorithms and Models)

כל אחת מהטכניקות שהוצגו עד כה נתמכת על ידי מספר אלגוריתמים ומודלים מתמטיים שונים. לדוגמה, קיימים מספר אלגוריתמים שונים המאפשרים הסקת חוקים ויחסים בין קבוצת פרמטרים. קיימים מספר רב של אלגוריתמים המסוגלים לבנות עצי החלטות, על בסיס קובץ מופעים גדול - CART (Classification and Regression Trees), CHAID (Chi-Squared Automatic Interaction Detection), C4.5 ו-C5.0 ועוד.

רוב האלגוריתמים והמודלים העומדים בבסיס הגישות השונות, התפתחו כתוצאה ממאמצי מחקר בתחומי זיהוי תבניות (Pattern Recognition) ואינטליגנציה מלאכותית. חלק מהאלגוריתמים הם קנייניים, וידועים רק לחברה שפיתחה אותם, בעוד שחלק אחר מהאלגוריתמים ידועים לכל והם סטנדרטיים.

מוצרים לכריית נתונים (Data Mining Tools)

שוק מוצרי כריית הנתונים נמצא עדיין בתהליך התהוות. לכן, פועלות בו מספר רב של חברות קטנות. השחקנים הגדולים מתחילים להיכנס לשוק באמצעות פיתוח עצמי, רכישת חברות, בניית ממשקים למוצרים אחרים וגם רכישת טכנולוגיות מחברות אחרות ושילובן במוצרים שלהם. להלן מספר דוגמאות לגבי המתרחש בשוק המוצרים.

- Silicon Graphics פיתחה מוצר בשם MineSet, המכיל מיגוון רחב של אלגוריתמים לוויזואליזציה, עצי החלטות ועוד.
- IBM פיתחה מוצר בשם Inteligent Miner, המכיל כלים מבוססי רשתות עצביות, כלים להסקת עצי החלטות, אלגוריתמים סטטיסטיים וכלי ויזואליזציה.
- Thinking Machines Corp., פיתחה את המוצר Darwin, המשלב אלגוריתמים לרשתות עצביות, אלגוריתמים גנטיים, k-nearest neighbor, Classification and Regression Tree להפעלת תהליכי סיווג וחיזוי.
- Business Objects הוסיפה מודל בשם BusinessMiner למחולל השאילתות הפופולרי שלה, ושילבה אותו יחד עם מודול OLAP, לניתוח רב-מימדי. זהו מוצר עם דגש חזק לצד תחנת העבודה של המשתמש, ולכן אינו מתאים לכריית נתונים בנפחים גדולים וברמת מורכבות גדולה. המוצר תומך בטכניקות גילוי חוקים ויחסים, עצי החלטות, פילוח אוכלוסיות ועוד.
- NeuralWare פיתחה מוצר בשם NeuralWorks Predict, המשמש לחיזוי על בסיס רשתות עצביות, סטטיסטיקה, אלגוריתמים גנטיים ולוגיקה עמומה. המוצר מכיל רכיב לבניית הדוגמאות בשלב הלימוד (Build Wizard) ורכיב Explain, המנסה להסביר את התוצאות המתקבלות.
- NCR פיתחה את המוצר Knowledge Discovery Workbench.
- DataMind Corp. פיתחה מוצר לכריית נתונים וחתמה על הסכם שיתוף פעולה עם Arbor Software Corp., המפתחת של כלי הניתוח הרב-מימדי Essbase.

❖ Pilot Software Inc. שילבה במוצר הניתוח הרב-מימדי שלה טכנולוגיית עצי החלטות. המוצר מאפשר הפעלת טכניקות עצי החלטות לפילוח הלקוחות ולהזנת הפלחים שאותרו להיררכיית המימדים של המנוע הרב-מימדי.

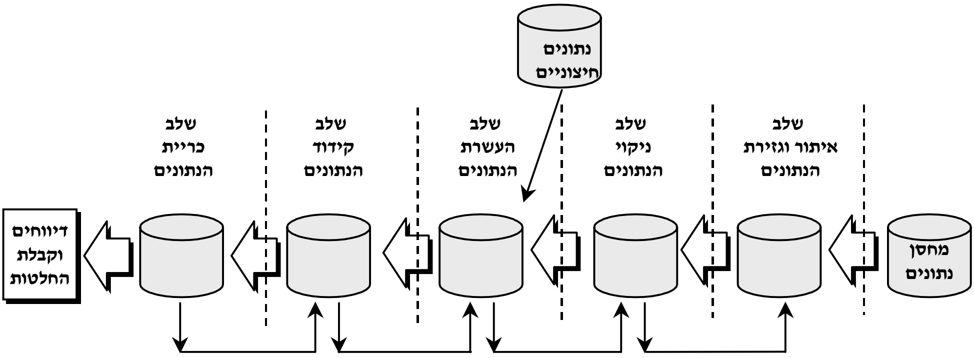
❖ SPSS, המייצרת את החבילה הסטטיסטית הנפוצה, שילבה במוצר שלה אלגוריתמים לרשתות עצביות וטכניקות Chi Square Automatic Interactive Detection - CHAID. חלק מהחברות שילבו במוצר שלהן מספר טכניקות שונות ומאפשרות למוצר לבחור מהי הטכניקה המתאימה, בהתאם להגדרת הבעיה.

חלק מהמוצרים עדיין מתמקד בעיקר בצד האלגוריתמי, ומזניח בדרך כלל את ההיבטים האחרים (ממשק משתמש, קלות שימוש, מבנה נתונים לקלט, ביצועים, תפעול הכלי וכדומה). רוב הכלים דורשים הכנת הנתונים וגזירתם למבנה נתונים שטוח ופשוט, ולא מעט עיבודים גם של התוצאות. יחד עם זאת, ככל ששוק כריית הנתונים גדל, משתכללים גם המוצרים, וניתן למצוא כבר מוצרים בשלים.

רוב המוצרים משתמשים ב-Windows, או בדפדפן אינטרנט, כממשק הגרפי ומאפשרים גישה לבסיסי הנתונים הטבלאיים באמצעות ODBC. חלק מהמוצרים המתקדמים (כגון Darwin של חברת Thinking Machines Corp.) מתחילים לנצל את הארכיטקטורות המקביליות של החומרה, לשיפור הביצועים.

תהליך כריית הנתונים

כריית נתונים אינה **פעולה**, אלא **תהליך** מתמשך, ובדרך כלל גם לא חד פעמי. נציג את השלבים השונים בתהליך כריית הנתונים. בדומה לחלוקת המאמץ, בכל הקשור להקמת מחסני הנתונים, גם כאן הסטטיסטיקה מצביעה על 80% מאמץ בהכנת הנתונים ו- 20% מאמץ בתהליך כריית הנתונים עצמו.



תרשים 7.10: תהליך כריית נתונים.

בעיקרון, תהליך כריית הנתונים מורכב משלבים אלה:

- ❖ **הגדרת מרחב הבעיה** (Problem Formulation) - תהליך כריית הנתונים מתחיל בהגדרת הבעיה. כלומר, מהו הידע שברצוננו לקבל ומה מטרת קבלת ידע זה. לדוגמה, בנק יכול להחליט שהוא רוצה להבין טוב יותר את סוגי לקוחותיו במטרה לשלוח לכל אחד מהם מכתבים אישיים (Personalized letters). תוכן המכתב האישי נקבע בהתאם לקבוצת הלקוחות אליה משתייך הלקוח. בשלב זה, עלינו לשאול את עצמנו גם שאלות ראשוניות כמו "כמה אשכולות אנו רוצים למצוא". אם הצורך הוא לארגון מחדש של מחלקת השיווק לפי סוגי הלקוחות, סביר להניח שנרצה מספר קטן של אשכולות גדולים (ארבעה עד חמישה), אבל אם מטרתנו היא לשלוח מכתבים אישיים, ייתכן שנרצה לעדן את הקבוצות ולקבל מספר עשרות אשכולות שונים. במקביל יש לוודא את יכולת הארגון לנצל את תוצאות תהליך כריית הנתונים. לדוגמה, אם מחלקת ההפצה אינה ערוכה להדפיס מכתבים אישיים ולהפיצם, מה התועלת בבניית אשכולות הלקוחות? מכיון שתהליך כריית הנתונים הוא מורכב ויקר, עדיף לשאול את כל שאלות אלו לפני, ולא לעמוד נבוכים לאחר שכל המאמץ נעשה.

- ❖ **הכנת סביבת העבודה** (Data Mining Environment Setup) - לפני שמתקדמים לשלבים הבאים עלינו לוודא את מוכנות המרכיבים השונים בסביבת העבודה לתהליך כריית הנתונים. בשלב זה עלינו לוודא את זמינות החומרה, התוכנה, רשת התקשורת, כלי כריית הנתונים וכדומה. יש לחשב את ההשקעות הנדרשות להקמת סביבת כריית נתונים כהשקעה ארוכת טווח, ולעשות כל מאמץ לבסס סביבה זו על סביבת מחסן הנתונים.

- ❖ **איתור וגזירת הנתונים** (Data Selection) - לאחר שהגדרנו את מרחב הבעיה ואת מטרת התהליך, והשתכנענו ביכולת לעשות שימוש בתוצאות התהליך, מתחיל שלב איתור וגזירת הנתונים. שלב זה יהיה קל באופן משמעותי אם הארגון כבר מפעיל מחסן נתונים, לעומת מצב בו יש לאתר ולגזור את הנתונים מאוסף רב של מערכות תפעוליות. בסביבת מחסן נתונים, סביבת כריית הנתונים היא למעשה מחסן משנה (Data Mart) אותו יש לאכלס ולתחזק באופן שוטף. בשלב זה מתבצע מיפוי של הנתונים הקיימים, מתבצעת הערכה בדבר איכותם וזמינותם, מכינים את תוכניות הגזירה ממחסן הנתונים או מהמערכות התפעוליות, ותוכניות הטעינה למבנה הנתונים הנדרש על ידי כלי כריית הנתונים.

- ❖ **ניקוי הנתונים** (Data Cleaning) - שלב זה מיועד לניקוי ושיפור טיב הנתונים, בעיקר עבור ארגון המבקש להפעיל תהליך כריית נתונים ללא מחסן נתונים תומך. ההנחה הבסיסית היא שקיומו של מחסן הנתונים מאפשר להניח שטיב הנתונים גבוה. בכל מקרה, חובה לבחון את איכות הנתונים, גם אם ניתן להתבסס על מחסן נתונים. לדוגמה, ייתכן שמחסן הנתונים מכיל רשומות לקוח ללא תאריך לידה, מכיון שנתון זה לא הוגדר הכרחי. אם נרצה לבסס ניתוח לקוחות גם על פרמטר גיל הלקוח, לא נוכל לעשות זאת ללא תאריך הלידה שלו. בהחלט ייתכנו מצבים שכתוצאה מפרויקט כריית נתונים יידרשו שינויים במערכות תפעוליות (למשל,

הפיכת שדה תאריך הלידה לשדה חובה) ושינויים מסוימים בתהליכי טעינת מחסן הנתונים (למשל, אם עד עכשיו נתון תאריך הלידה לא נוהל בכלל במחסן הנתונים).

- ❖ **העשרת הנתונים** (Data Enrichment) - בשלב זה נעסוק בהעשרת הנתונים, במידת הצורך. לדוגמה יכול להיות שנוסיף לכל רשומה נתונים דמוגרפיים מסוימים ממאגר חיצוני שנרכוש במיוחד למטרה זו - למשל, רמת ההכנסה הממוצעת באזור מגורים מסוים, מחזור המכירות של חברות וכדומה. לא תמיד קל להפגיש מידע חיצוני עם נתונים פנימיים של הארגון, וייתכן שנצטרך לבנות תהליכי הפגשה מתוחכמים.

- ❖ **קידוד הנתונים** (Data Coding) - בשלב זה יש להחליט כיצד אנו רוצים לייצג את הנתונים. לדוגמה, נתונים רציפים (כגון רמת הכנסה) ניתן לייצג על ידי טווחים, נתונים מסוימים ניתן לייצג על ידי קודים, וכדומה. למשל, מוצרים מסוימים, המבוססים על אלגוריתמים של רשתות עצביות (כפי שתיארנו בסעיפים הקודמים), יכולים לקבל כקלט רק ערכים נומריים. אם חילקנו את הארץ ל-20 אזורים גיאוגרפיים ולכל אחד קוד אלפאנומרי, יהיה צורך לבנות רשומה ובה 20 שדות בינאריים, שכל אחד מהם מכיל 0 או 1, בהתאם לאזור הגיאוגרפי של הלקוח. תהליך בניית הקודים תלוי, לא מעט, בדרישות וביכולות הכלים לכריית נתונים.

- ❖ **כריית הנתונים** (Data Mining) - זהו לב תהליך כריית הנתונים וכאן מופעל למעשה תהליך הגילוי, שאינו בהכרח השלב המורכב ביותר, כפי שניתן להבין מהשלבים הקודמים. מסתבר שהכנת הנתונים לכרייה הוא תהליך מורכב מאוד הדורש השקעת מאמץ רב. בשלב זה יש להחליט באיזו גישה להשתמש, איזה אלגוריתם ספציפי יופעל, וכל זאת בהתבסס על הכלים שנרכשו או זמינים בארגון. בדרך כלל, מתחילים את תהליך הכרייה בהפקת מספר שאילתות או דוחות על הנתונים, כדי לרכוש קצת יותר הבנה בנתונים. לאחר שלב ראשוני זה מתחילים בהפעלת הכלים הייעודיים.

- ❖ **דיווח** (Reporting) - זהו השלב בו מנתחים את ממצאי תהליך כריית הנתונים, מסכמים את התוצאות, מגבשים את דרך הצגתם ומציגים אותם למקבלי ההחלטות.

התהליך המתואר כאן אינו תהליך לינארי חד כיווני, אלא יכול להיות תהליך איטרטיבי - למשל בחירת נתונים מחדש לאחר תהליך כריית הנתונים (בשל גילוי עובדות המחייבות פרמטרים בהם לא בחרנו, למשל זיהוי נתונים לא נקיים תוך כדי תהליך הכרייה וכדומה). בנוסף, תהליך הניתוח יכול לשלב גם טכניקות אנליטיות נוספות, מעבר לכריית נתונים. לדוגמה, כתוצאה מכריית נתונים ייתכן שנחליט לבצע ניתוח רב-מימדי באמצעות כלי OLAP כלשהו, או להפיק אוסף שאילתות באמצעות מחולל שאילתות כלשהו. חשוב להסתכל על כל הטכניקות האנליטיות כעל טכניקות משלימות העומדות לרשות המשתמשים, ולא כטכניקות המוציאות זו את זו.

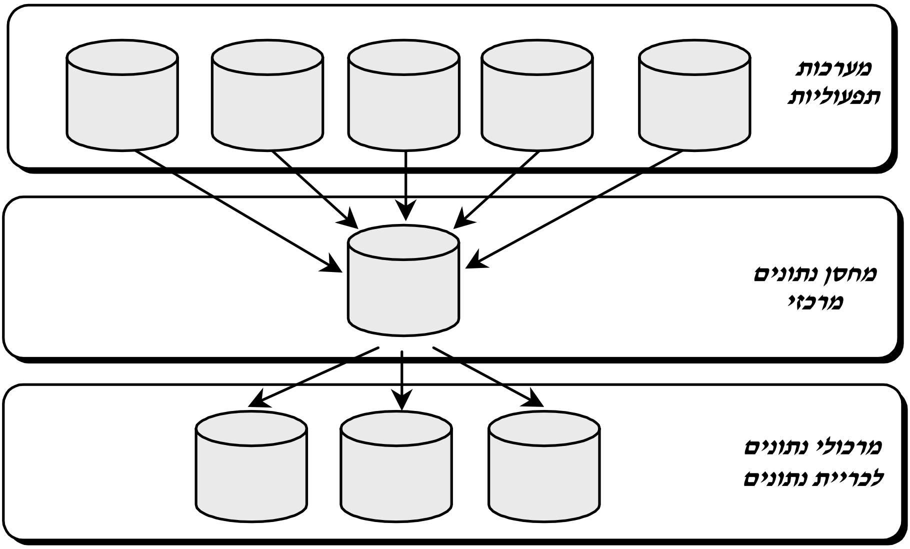
אינטגרציה בין מחסן הנתונים לכריית הנתונים

כל תכלית קיומו של מחסן הנתונים בארגון הוא תמיכה בתהליך קבלת ההחלטות. לעיתים, המידע שניתן להעמיד לרשות מקבלי ההחלטות באמצעות כלי שאילתות מבוססי SQL, או כלי ניתוח רב-מימדי, מספיקים בהחלט. הבעיה מתחילה ברגע שקיים קושי להגדיר בדיוק מה אנו מחפשים בים הנתונים האגור במחסן הנתונים, וקיימת הבנה שמספר יהלומים עדיין מסתתרים בו. זהו הרגע בו מחליט הארגון להתחיל בתהליך כריית הנתונים.

כפי שכבר צריך להיות ברור מהדיון בפרק זה, קיומו של מחסן הנתונים אינו תנאי קדם לשימוש בטכניקות כריית נתונים. אולם, מאידך, קיומו יכול לפשט ולהאיץ באופן משמעותי את יכולת הארגון לנצל את כריית הנתונים לאיתור אותם יהלומים. ארגון יכול להגיע למסקנה שכריית נתונים יכולה לסייע לו בתהליך קבלת החלטות, ולהתחיל ביישום הנושא, ללא קיומו של מחסן נתונים. הארגון יעשה זאת על ידי גזירת נתונים ישירות ממערכות תפעוליות. אין כל ספק שתהליך זה מורכב בהרבה, צורך משאבים גדולים יותר ולעיתים אף מסכן את איכות התוצאות. מכל האמור לעיל נובע שכריית הנתונים היא תפיסה המשתלבת באופן טבעי בתפיסת מחסן הנתונים, ומומלץ מאוד לראות את שני הנושאים כנושאים אינטגרטיביים.

בדרך כלל, כריית הנתונים תעשה במחשב נפרד ממחשב מחסן הנתונים, וזאת בשל משאבי המחשב הגדולים שדורשות טכניקות כריית נתונים מסוימות. הפעלת טכניקות אלו באופן ישיר במחשב מחסן הנתונים אפשרית, אולם יכולה לגרום לירידה משמעותית בזמני התגובה של שאילתות הפועלות בסביבת מחסן הנתונים. למרות שחלק ממוצרי כריית הנתונים פועלים ישירות על בסיסי נתונים טבלאיים מסחריים רגילים, עדיין קיים הצורך בתהליך הכנת הנתונים במבנה מיוחד (למשל, על ידי הפיכת שדות ערכים לשדות טווחים, העברת שדות מסוימים למבנה קודים כגון קיים/לא קיים וכדומה). נדגיש, שקיימים גם מוצרי כריית נתונים שאינם מסוגלים לפעול באופן ישיר מול בסיס נתונים טבלאי, אלא דורשים מבנה נתונים ייעודי, כמו למשל קובץ שטוח.

בשל דרישות משאבי המחשב, וכן בשל הצורך לבצע תהליכי הכנת נתונים לקראת כריית הנתונים, הדרך הנכונה ביותר היא להתבסס על תפיסת מרכול הנתונים. כלומר, לבנות מחסן נתונים ייעודי לכריית הנתונים.



תרשים 7.11: מרכולי נתונים לכריית נתונים.

בעיות עיקריות בכריית הנתונים

כריית הנתונים אינה פעילות העומדת בפני עצמה. אחת מדרישות הקדם העיקריות היא קיומו של מחסן נתונים. נסקור בקצרה את הבעיות העיקריות בתהליך כריית הנתונים.

- **רגישות גבוהה לאיכות הנתונים** - שימוש בכריית נתונים על נתונים לא אמינים, לא שלמים ולא איכותיים, עלול להפיק תוצאות חסרות משמעות, או במקרה הגרוע יותר - להחלטות שגויות. לצערנו, גם בשטח זה תופס העיקרון "נכנס זבל, יצא זבל" המוכר כל כך משטחי מחשוב אחרים. חובה להשקיע את המאמץ הנדרש בתהליך ניקוי הנתונים, ולכן התבססות על נתונים שמקורם במחסן הנתונים (ולא במערכות התפעוליות) מהווה נקודת התחלה טובה לפרויקט כריית נתונים.

- **העדר נתונים מספיקים** - בסיסי נתונים של מחסני הנתונים חסרים לעיתים את התכונות והמאפיינים הדרושים לתהליכי כריית הנתונים. הבעיה היא שלעיתים נתונים אלה פשוט אינם קיימים במערכות התפעוליות, וגם השגתם ממערכות חיצוניות אינו פשוט. לדוגמה, אם נתוני ספירת כדוריות האדומות בדם אינם ידועים, אין אפשרות להשתמש בתהליך כריית נתונים כדי לחזות למי מלקוחות קופת החולים יש הסתברות גבוהה יותר לחלות במחלה מסוימת.

- **העדר היכולת להסביר את התוצאות** - רוב הכלים אינם יכולים להסביר את התוצאות אליהן הגיעו. הכלים מבוססים על אלגוריתמים מתוחכמים, וקיים קושי לא קטן להסביר כיצד הגיע האלגוריתם לתוצאה. בעיה זו היא מהותית, מכיון שקיים קושי להסביר למשתמשים את דרך ההגעה לתוצאות, ולכן גם קיים חשש אצל המשתמשים לבסס את החלטותיהם על התוצאות.

- ❖ **מבנה נתונים ייחודי** - חלק מהכלים אינם פועלים באופן ישיר על בסיסי נתונים טבלאיים, אלא זקוקים למבני נתונים מיוחדים (בדרך כלל, קבצים שטוחים פשוטים). הצורך לגזור את הנתונים ממחסן הנתונים, ולהמיר אותם למבנה מיוחד, גורם לסיבוך נוסף בתהליך כריית הנתונים. מכיון שמבנה הכוכב של מחסן הנתונים יכול להכיל רמה גבוהה של דה-נורמליזציה, צריך למפות את מבנה הכוכב למבנה שטוח. תוך כדי תהליך כריית הנתונים מתגלות עובדות לא ידועות הדורשות חקירה נוספת על פרמטרים שלא נגזרו, וצריך לטעון מחדש את הנתונים פעם נוספת.
- ❖ **ביצועים** - חלק מהכלים מוגבלים בגודל הקבצים שהם מסוגלים לקבל כקלט. מכיון שבדרך כלל תהליך כריית הנתונים מבוסס על Single Thread, יש קושי לנצל את המבנה המקבילי של המחשב, כדי להגיע לביצועים טובים יותר. למרות שחלק מהכלים המודרניים מתחילים לתמוך בגישה ישירה לבסיסי הנתונים הטבלאיים הפופולריים (בסיסי נתונים התומכים במקביליות), כלי כריית הנתונים עצמם אינם תומכים במקביליות. חלק מהיצרנים מנסים לעקוף את בעיית הביצועים על ידי המלצה לעבוד עם קובץ המכיל מדגם כלשהו. בניית מדגם כשלעצמה אינה תהליך טרוויאלי. הוא עלול להטעות את האלגוריתמים, וכמובן את התוצאה המתקבלת.
- ❖ **קושי להצדיק את עלות הפרויקט** - קיים קושי לבנות את מודל העלות/תועלת של תהליך כריית הנתונים. לעיתים, דורש תהליך כריית הנתונים חומרה נפרדת, מהנדסי תוכנה לבניית תהליכי הכנת הנתונים, מומחים שעברו הכשרה בכלים ובעלי ידע מתקדם בסטטיסטיקה ובשיטות מתמטיות, ורכישת כלי תוכנה ייעודיים. כל אלה עולים כסף רב, ויש קושי להתחייב מראש לגבי תוצאות התהליך, איזה עובדות בלתי ידועות נצליח לגלות ומה תהיה משמעות גילוי זה, מבחינה עסקית.

סיכום

תפיסת כריית הנתונים יוצאת ממעבדות המחקר ונכנסת במהירות למרכזי המחשוב הגדולים. היא תופסת מקום של כבוד על במת המערכות לתמיכה בתהליכי קבלת החלטות ומחסני הנתונים. ניתן לומר שזו מגמה ברורה, שרק תגבר בשנים הקרובות. הלחצים התחרותיים דורשים ניתוח מהיר ומתוחכם של הנתונים, וכריית הנתונים היא כיום ללא ספק פסגת התהליכים האנליטיים. ככל שבסיסי הנתונים יגדלו מבחינת נפח ומורכבות, הצורך בכלים לכריית נתונים ילך ויגדל. זאת בשל הקושי ההולך וגדל של המוח האנושי להתמודד עם הבנת וניתוח הנתונים. למרות שיישומי כריית הנתונים אינם מתאימים לכל סוגי הבעיות, מוצרי כריית הנתונים כבר מספקים תוצאות מלהיבות בכל הקשור לפילוח לקוחות, איתור הונאות וניתוח סל הקניות. למרות שלא כל חברה זקוקה כיום לטכנולוגיה זו, סביר להניח שבזמן זה או אחר רוב החברות תשתמשנה בה, או בנגזרת כלשהי שלה. טכנולוגיה זו משלימה את הטכניקות האנליטיות האחרות, כגון מחוללי השאילתות ומוצרי OLAP, ומוסיפה מימד חדש ליכולת ניתוח מידע.

פרק 8: הטכנולוגיה המקבילית בשירות מחסן הנתונים

מבוא

מטרת פרק זה לסקור כיצד תומכת טכנולוגיית העיבוד המקבילי (Parallel Processing) בדרישות הייחודיות לגבי ביצועים ונפחי הנתונים בסביבת מחסן הנתונים. הבסיס לעיבוד המקבילי נובע מארכיטקטורות חומרה מיוחדות המאפשרות עיבוד מקבילי על ידי שימוש במספר מעבדים. על מצע הארכיטקטורה המקבילית של החומרה פועלת התוכנה. זו מנצלת את ריבוי המעבדים, כדי לחלק משימה אחת למספר משימות אותן ניתן לבצע במקביל, וכך לקצר את משך ביצוע המשימה הכולל. מערכות RDBMS המודרניות מסוגלות לקחת שאילתת SQL אחת ולחלק אותה למספר משימות המתבצעות במקביל. כדי לאפשר למשימות לפעול במקביליות רבה ככל הניתן פותחה שיטה לחלוקת טבלה גדולה למספר **מחיצות** (Partitions). בדרך זו ניתן לנצל מעבדים שונים כך שיסרקו במקביל קטעים שונים באותה טבלה.

כפי שנראה, ניתן להפעיל את העיבוד המקבילי במיגוון רחב של משימות. ביניהן:

- סריקת טבלאות (Table Scan) וסריקת אינדקסים (Index Scan)
- מיון מקבילי של תוצאות השאילתות (Table Sort)
- ניתוח מקבילי של שאילתות (Query Parsing)
- עיבוד מקבילי של פעולות צירוף טבלאות (Join)
- זרימה מקבילית של מספר שאילתות שונות (Pipelining)
- עדכון מקבילי של טבלה
- טעינה (Load) וגיבוי (Backup) של טבלאות

מערכות מסוימות תומכות בעיבוד מקבילי של חלק מהמשימות שהזכרנו, בעוד שאחרות מסוגלות לתמוך בעיבוד מקבילי של כולן. תמיכה בעיבוד מקבילי עבור חלק מהמשימות בלבד עלול להיות בעייתי (למשל, תמיכה בסריקה מקבילית של טבלאות. עם זאת, העדר תמיכה בטעינה מקבילית של טבלאות עלולה להוות צוואר בקבוק משמעותי). מכאן החשיבות בהבנת המושגים השונים ובחינת המערכות המסחריות מנקודת מבט של תמיכתן בעיבוד מקבילי.

ארכיטקטורות מרובות מעבדים לעיבוד מקבילי (Parallel Systems)

לפני שנתייחס לתכונות בסיס הנתונים, נסקור תחילה מספר ארכיטקטורות חומרה המבוססות על ריבוי מעבדים, ותומכות בעיבוד מקבילי. המושג **עיבוד מקבילי** הוא רחב ויכול להתייחס, בקצהו הנמוך, לשני מעבדים הפועלים בשרתי Intel ועד למערכות בהן יש מאות ואלפי מעבדים. השוני בין המערכות השונות נובע יותר מארכיטקטורת המערכת, מאשר מהעוצמה של המעבד הבודד.

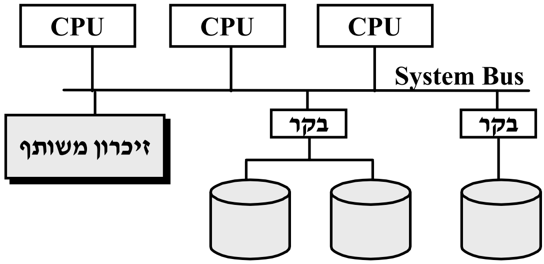
מטרת הדיון היא להציג באופן כללי ביותר את הארכיטקטורות, ונועד אך ורק לשמש כבסיס לדיון בשיטות העיבוד המקביליות של בסיסי נתונים בסביבות מחסני נתונים.

ארכיטקטורת SMP (Symmetric Multi Processing)

בארכיטקטורה זו אוסף של מעבדים משתפים ביניהם זיכרון (Shared Memory) ודיסקים. בשל שיתוף הזיכרון והדיסקים בין המעבדים מקובל גם לקרוא לארכיטקטורה זו בשם Shared Everything Architecture. זו ארכיטקטורה סימטרית, המתבטאת בעובדה שכל המעבדים שווי זכויות בגישתם לזיכרון והדיסקים. שיתוף הפעולה בין המעבדים מתבצע באמצעות הזיכרון המשותף. גישת המעבדים לזיכרון נעשית באמצעות אפיק (bus) מהיר, או התקן מיתוג מהיר (switch based interconnect).

יש תחרות בין המעבדים ויחידות קלט/פלט על הגישה למשאב המשותף - הזיכרון. כדי למנוע התנגשות דרוש סינכרון בין המעבדים ותיאום מתמיד ביניהם, ולכן יש גבול למספר המעבדים שניתן להוסיף ועדיין לקבל שיפור בביצועי המערכת. החיסרון העיקרי של ארכיטקטורה זו הוא כושר הגידול שלה. עד מספר מסוים של מעבדים הגידול בעוצמת המערכת הוא יחסי לעוצמת המעבדים; מעבר לנקודה זו הגידול בעוצמת המערכת קטן באופן משמעותי לעומת תוספת המעבד. צווארי הבקבוק נוצרים מכיון שמעבדים רבים מתחרים על אותם משאבים. מספר המעבדים יכול לנוע, בדרך כלל, בין שני מעבדים ועד 16 מעבדים. פותחו גם מערכות בהן ניתן להתקין 36, 64 ואפילו 96 מעבדים, תוך השגת שיפור בביצועים. עם זאת, ברוב המערכות מותקנים בין 12 ל-16 מעבדים.

אחד הפתרונות המשפרים את ביצועי מערכת SMP הוא שימוש בזיכרון מטמון (Cache Memory) עבור כל מעבד. זיכרון המטמון מאפשר לכל מעבד לקבל שטח זיכרון מסוים משלו, כך שלא כל הזיכרון משותף לכלל המעבדים. יצרנים פועלים גם להגדיל את עוצמת ערוץ התקשורת של מערכות SMP, כך שיותר נתונים יכולים לעבור בין רכיבי המערכת השונים. בנוסף להגדלת מספר המעבדים, ניתן גם להחליף מעבדים חלשים במעבדים חזקים יותר. לכן, עוצמת העיבוד אינה נמדדת **רק** על פי מספר המעבדים.



תרשים 8.1: ארכיטקטורת SMP.

אחד מיתרונותיה העיקריים של ארכיטקטורה זו מתבטא בכך שהיא קלה לניהול ותחזוקה, מכיון שהיא מספקת תמונת מערכת אחת (Single Image), מנקודת מבטה של מערכת ההפעלה ומערכות התוכנה הפועלות בה. בדרך כלל, אין צורך לשכתב יישומים כדי לאפשר להם לפעול בסביבת SMP. הגידול בעוצמת עיבוד המערכת הוא שקוף, ואינו דורש שינוי כלשהו ביישומים.

להלן טבלה המציגה את היצרנים הנפוצים ביותר של מערכות SMP:

שם יצרן	שם מערכת
Compaq (Digital)	AlphaServer 8200, 8400
HP	HP 9000
IBM	RS/6000
NCR	3500, Worldmark 5100
Pyramid Technology	Nile Series
Sequent Computer Systems	Symetry 5000
Silicon Graphics	Chalenge, CS6400, Onyx
Stratus Computer	Continum
Sun Microsystems	SparcServer

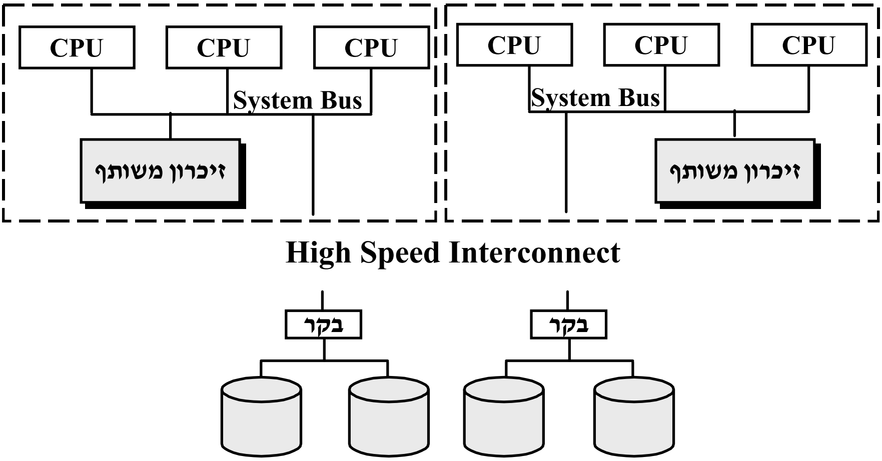
ארכיטקטורת אשכול (Cluster Architecture)

בארכיטקטורה זו מתקינים מספר מחשבים שונים, שכל אחד מהם יכול להיות בארכיטקטורת מעבד בודד או בארכיטקטורת SMP, ומשתפים ביניהם דיסקים. כל מחשב משתמש בזיכרון ובמערכת ההפעלה שלו, אולם יש לו גם זכות גישה לזיכרון במחשבים האחרים. מכיון שלכל המחשבים זכויות גישה שוות (peer-to-peer access) לזיכרון המחשבים האחרים, מקובל לקרוא לארכיטקטורה זו **זיכרון משותף באשכול**. בדומה, מכיון שכל המחשבים משתפים ביניהם אוסף משותף של דיסקים, ניתן לקרוא

לארכיטקטורה זו גם **ארכיטקטורת דיסק משותף** (Shared Disk). התקשורת בין המערכות מתבצעת על ידי העברת מסרים (message passing) דרך ערוץ תקשורת מהיר (High Speed Interconnect). מטרת האשכול היא ליצור תחושה של מערכת גדולה וחזקה יותר.

להבדיל מארכיטקטורת SMP רגילה, מערכת ההפעלה הסטנדרטית אינה יודעת לפזר, באופן אוטומטי, את עומס העיבוד על צמתים שונים באשכול. מערכות ההפעלה יודעות לחלק את עומס העיבוד בצומת SMP, אך לא בין צמתים שונים. לכן, נדרש תחכום מיוחד של מערכת ההפעלה כדי לתמוך בארכיטקטורת האשכול. שכבת תוכנה מיוחדת פועלת בכל צומת של האשכול ומנהלת את התקשורת בין המחשבים השונים. היא עושה זאת באמצעות הערוץ המהיר ותפקידה להביא לתמונה עקבית של זיכרון המטמון (Cache Coherency) ושל הזיכרון הרגיל.

מכיון שהדיסק משותף בין מספר מחשבים, נדרש מנגנון נעילה מיוחד לניהול נעילות מבוזרות (Distributed Lock Manager). תפקיד מנגנון זה להבטיח שהנתונים בדיסק יעודכנו רק על ידי צומת אחת. כמו כן, מכילה שכבת התוכנה מנגנון לניהול תורים, המאפשר למחשבים באשכול לעבוד מבלי לפגוע באמינות ושלמות הנתונים בדיסק. בשל התחכום הנדרש לניהול האשכול, פותחו מערכות אלו על ידי יצרנים שפיתחו את החומרה, את מערכות ההפעלה ואת מנגנוני התקשורת בין הצמתים. ארכיטקטורה זו אינה מחייבת שכל המעבדים יהיו מאותו סוג, אלא רק שיפעלו עם אותה מערכת הפעלה, ולכן ניתן לשתף מספר מחשבים בעלי מעבד שונה.



תרשים 8.2: ארכיטקטורת אשכול.

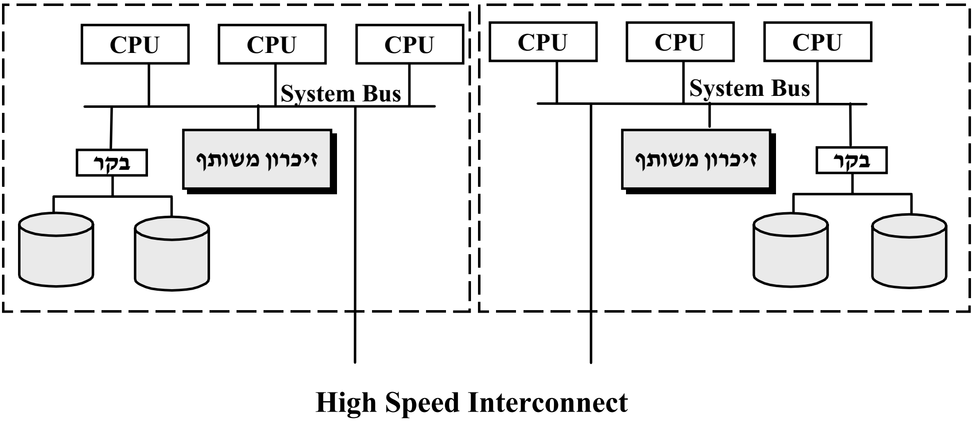
בנוסף ליכולת העיבוד המקבילי של הפעלת משימות שונות על מעבדים שונים, מאפשרת ארכיטקטורה זו רמה גבוהה של זמינות, מכיון שנפילת מחשב אחד מאפשרת המשך מתן השירות על ידי המחשבים הנותרים. ארכיטקטורה זו מאפשרת גם איזון עומסים טוב, מכיון שניתן להחליט איזו משימה תתבצע על ידי איזה מחשב. גם בארכיטקטורה זו קיימת מגבלה על מספר המחשבים שניתן לשתף מול מערכת דיסקים אחת. ככל

שמוסיפים מערכות לאשכול, הצורך בתאום בין המערכות והעומס על ערוץ התקשורת המהיר, הולך וגדל ולכן, התועלת מתוספת עוצמת העיבוד מתחילה לרדת. שיפור אפקטיבי בעוצמת העיבוד נקבל באשכול המורכב מ-4 עד 8 צמתים. חברת דיגיטל (כעת Compaq) היתה בין החלוצות בפיתוח ארכיטקטורה זו במערכות VAX/VMS שלה, בתחילת שנות ה-80. גם חברת Tandem השתמשה בארכיטקטורה זו כדי לבנות מערכות בעלות רמות זמינות גבוהות מאוד. כיום ארכיטקטורה זו נתמכת על ידי רוב יצרני מערכות Unix.

ארכיטקטורת MPP (Massively Parallel Processing)

ארכיטקטורה זו מבוססת על אוסף מעבדים, כאשר לכל מעבד יש את העותק שלו של מערכת ההפעלה, את הזיכרון שלו ואת אוסף הדיסקים שלו. לכן, מקובל לומר שזו ארכיטקטורה בה המעבד אינו משתף שום משאב (Shared Nothing). חלק מהיצרנים מאפשרים שיתוף דיסק אחד לשני צמתים, כדי למנוע מצב של השבתת המערכת (אם צומת אחד נופל). הארכיטקטורה מאפשרת עצמאות של כל מעבד, יחד עם תקשורת ושיתוף פעולה ביניהם. התקשורת בין המעבדים מתבצעת על ידי העברת מסרים באמצעות ערוץ תקשורת מהיר, בפרוטוקולים מיוחדים המפותחים על ידי היצרנים. לארכיטקטורה זו כושר גידול עצום, וקיימות מערכות המגיעות למאות ולאלפי מעבדים. נדגיש, שמערכות אלו דורשות רמת תחכום בהחלטה לגבי פיזור הנתונים בין הדיסקים, כדי להגיע למינימום צורך בהעברת נתונים בין הצמתים. בדומה לארכיטקטורת האשכול, גם ארכיטקטורה זו זקוקה למנגנוני נעילות מבוזרים, כדי לשמור על אמינות ושלמות הנתונים, המפוזרים בין מספר דיסקים.

קיימת הסכמה כללית שמערכות אלו מורכבות יותר לניהול, ודורשות מאמץ ניכר הן בכתיבת היישומים ומערכות התוכנה שיפעלו בסביבה זו, והן בתחזוקה השוטפת של המערכות, באופן שיאפשר ליישומים לנצל את עוצמת המערכת.



תרשים 8.3: ארכיטקטורת MPP.

ארכיטקטורה זו מחייבת פיזור הנתונים בין הדיסקים השונים, כדי לאפשר למשימות שונות לרוץ באופן עצמאי בכל צומת. מערכות MPP מכילות מנגנון המבצע חלוקת טבלה למחיצות, המפוזרות על ידי החומרה לצמתים שונים (Data Striping). שיטה זו מוסברת בהמשך, בסעיף המסביר את שיטות החלוקה למחיצות.

אחת הסוגיות המעניינות בארכיטקטורה זו מתייחסת לשאלה מה עובר בין הצמתים השונים. בהנחה שיש משימה הפועלת בצומת מסוים, וזקוקה לנתונים הנמצאים בצומת אחר, ניתן לחשוב על שתי חלופות שונות:

❖ העברת הנתונים מהצומת בו הם מאוחסנים לצומת בו מתבצעת המשימה. לשיטה זו קוראים **שיגור נתונים** (Data Shipping). לדוגמה, אם יש לבצע פעולה של צירוף בין שתי טבלאות הנמצאות בשני צמתים שונים, יש לשגר טבלה מצומת אחד לצומת השני. שיטה זו יכולה להעמיס מאוד על ערוץ התקשורת, מכיון שטבלאות יכולות להיות גדולות מאוד.

❖ העברת המשימה מהצומת בו היא צריכה להתבצע לצומת בו נמצאים הנתונים. לשיטה זו קוראים **שיגור משימה** (Function Shipping). במקרה זה משוגר קוד מצומת לצומת, דבר שאינו מעמיס על ערוץ התקשורת. שיטה זו יעילה יותר, מכיון שהיא חוסכת שיגור של כמויות נתונים גדולות מצומת אחד למשנהו בערוץ התקשורת המהיר.

נדגיש, שבכל מקרה לא ניתן ליצור מצב בו מתבצע רק שיגור פונקציות ולכן, תמיד יש גם שיגור נתונים. על רכיב האופטימיזציה להבטיח את הקטנת שיגור הנתונים ולהשתמש בשיטה זו רק כאשר אין אפשרות אחרת.

להלן טבלה המציגה את היצרנים הנפוצים ביותר של מערכות MPP:

שם יצרן	שם מערכת
Fujitsu	AP 1000
HP	Exemplar SPP (Convex)
IBM	SP2
Intel	Paragon
NCR	1012, 3600, Worldmark
NCube	NCube-2, NCube-3
Pyramid Technology	Reliant RM1000
Silcon Graphics	T3E (Cray)
Compaq	Himalaya K10000, K20000

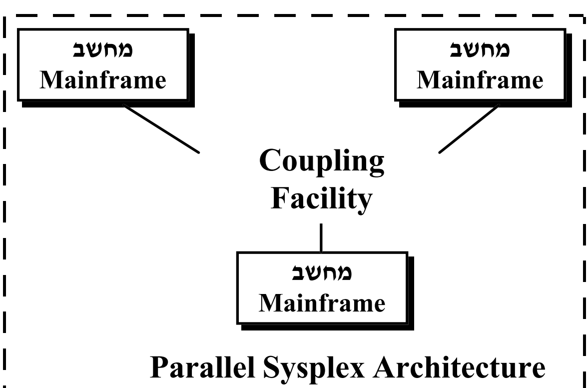
פיתוח מערכות MPP הואץ מאוד, כתוצאה מהופעת תפיסת מחסן הנתונים, בשל יכולתן לתמוך בשאילתות הפועלות מול בסיסי נתונים ענקיים ומשרתים מספר רב של משתמשים.

ארכיטקטורת NUMA
(Non Uniform Memory Architecture)

קיימת הסכמה כללית בין היצרנים השונים שעם הזמן נהיה עדים להופעת ארכיטקטורה המשלבת את יתרונות SMP עם יתרונות MPP. ארכיטקטורת NUMA, מייצגת שילוב מסוים של שתי תפיסות אלו, ומבוססת על אוסף מערכות SMP הקשורות ביניהן באופן הדוק (Tightly Coupled) באמצעות ערוץ תקשורת מהיר ביותר. ערוץ מהיר זה מקשר בין הזיכרון המקומי של צומת SMP אחד, לבין הזיכרון של צומת SMP אחר. תפיסת NUMA מבחינה בין הזיכרון הראשי, השייך לצומת מסוים, לבין הזיכרון המשני, שהוא הזיכרון של צומת מרוחק. מכיון שהערוץ מהיר ביותר, נוצרת תחושה של כמעט זיכרון מקומי, גם כשנדרשת גישה לנתונים המאוחסנים בצומת מרוחק. ארכיטקטורה זו מאפשרת למערכת הפעלה לפעול במצב הדומה לארכיטקטורת SMP, ולא כאוסף מערכות הפעלה הפועלות בנפרד בכל צומת, כפי שפועלת ארכיטקטורת MPP. מספר חברות (ביניהן Sequent, Data General, IBM ואחרות), פיתחו מערכות המבוססות על ארכיטקטורה זו. לדוגמה, המערכת של חברת Sequent מסוגלת לפעול עם 252 מעבדים שונים.

ארכיטקטורת Parallel Sysplex של חברת יבמ

חברת IBM פיתחה את ארכיטקטורת Parallel Sysplex. ארכיטקטורה זו מחברת אוסף של מחשבים מרכזיים, שכל אחד מהם מהווה צומת SMP, למערכת אחת כוללת (המזכירה את ארכיטקטורת NUMA). צומת SMP מבוסס על מחשב מרכזי הפועל בטכנולוגיית CMOS (או אף בטכנולוגיה Bipolar הישנה יותר) ויכול להכיל עד 10 מעבדים. ארכיטקטורת Parallel Sysplex מאפשרת שילוב של עד 32 צמתים כאלה, ולכן ניתן לקבל מערכת המורכבת מ-320 מעבדים. 32 הצמתים משתפים ביניהם את כל מערך הדיסקים.



תרשים 8.4: חיבור מחשבים מרכזיים.

רכיב מיוחד, WLM (Workload Manager), מפקח על המשאבים בצומת SMP, או באשכול המחשבים המחוברים ב- Parallel Sysplex. רכיב זה מפקח על המעבדים, ערוצי קלט/פלט והדיסקים, ומאזן את השימוש לקבלת ביצועים אופטמליים. רכיב זה מאפשר קבלת סביבת עבודה מאוזנת, כך שבאותה מערכת מופעלים גם יישומי OLTP וגם יישומי מחסן נתונים. ניתן לגרום לכך ששאילתות כבדות של מחסן נתונים לא יגרמו לירידה בביצועי יישומי OLTP. רכיב WLM מבטיח ששאילתה מסוימת של מחסן הנתונים לא תשתלט על כל משאבי המערכת.

עיבוד מקבילי של שאילתות (Parallel Query Processing)

מהו עיבוד מקבילי של שאילתה

הרעיון הבסיסי בעיבוד מקבילי של שאילתות פשוט להבנה, אם כי מורכב למימוש. הוא מבוסס על שני מושגים בסיסיים - חלוקת טבלה לוגית למספר מחיצות (Partitions) ויכולת עיבוד שלבים שונים של מספר שאילתות, בתפיסת צינור זרימה מקבילית (Pipelining). לפני שנסקור מושגים אלה נתבונן תחילה כיצד מתבצעת שאילתה בסביבת מעבד בודד. השאילתה הבאה מציגה את סך המכירות ללקוחות עסקיים שערך המכירה היה גדול מ-1000 ש״ח.

```
1. SELECT CUSTOMER_ID,
2.              SUM (SALE_QUANTITY * SALE_AMOUNT) AS SALE_TOTAL
3. FROM SALES S, CUSTOMERS C
4. WHERE C.CUSTOMER_ID = S.CUSTOMER_ID AND
5.              C.CUSTOMER_TYPE = 'Business' AND
6.              S.SALE_QUANTITY * S.SALE_AMOUNT > 1000
7.  ORDER BY  SALE_TOTAL
```

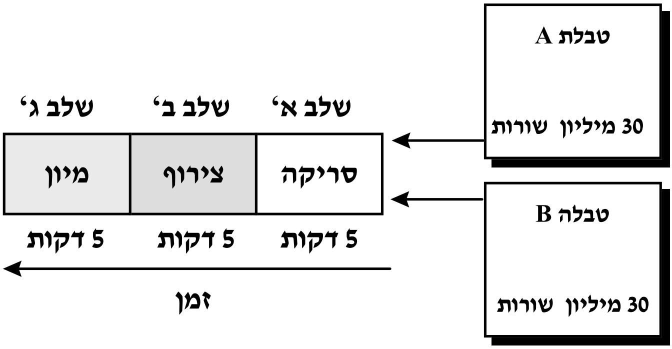
השלבים העקרוניים אותם צריכה השאילתה לבצע הם:

- שליפת שורות מטבלת המכירות, בהן ערך המכירה היה גדול מ-1000 ש״ח.
- שליפת שורות מטבלת הלקוחות, בהן סוג הלקוח הוא עסקי.
- צירוף שורות המכירות עם השורות מטבלת הלקוחות.
- מיון התוצאה בסדר יורד, לפי ערך יורד של המכירה.

בהפשטה, נוכל לומר שהשאילתה מורכבת משלבי הביצוע הבאים:

- סריקת (Scan) הטבלאות.
- צירוף (Join) תוצאות הסריקה.
- מיון (Sort) תוצאת הצירוף.
- הצגת תוצאת המיון למשתמש.

להבנת התהליך, נניח מספר הנחות פשטניות ביותר. למען הפשטות נניח ששתי הטבלאות מכילות 30 מיליון שורות כל אחת. במציאות, סביר להניח, שטבלת המכירות תהיה גדולה יותר מטבלת הלקוחות. כמו כן (שוב, למען הפשטות), נניח שזמן הביצוע של כל אחד משלושת שלבי השאילתה הוא 5 דקות. התרשים הבא מציג את שלושת שלבי השאילתה. במציאות כל שלב יכול לקחת זמן שונה, אולם הנחות אלו מיועדות רק להבנת העיקרון.

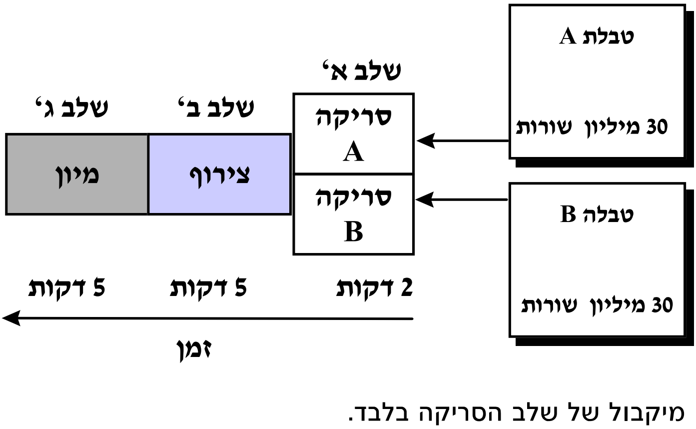


תרשים 8.5: ביצוע סדרתי של שלבי השאילתה.

מכיון שכל שלב לוקח 5 דקות לביצוע, סך כל זמן ביצוע השאילתה יהיה 15 דקות.

סריקה מקבילית של טבלאות (Parallel Table Scan)

עד כאן הנחנו שהשאילתה מתבצעת בחומרה עם מעבד בודד. כעת נניח שהשאילתה מתבצעת בחומרה מרובת מעבדים. במצב זה ניתן לחלק את השאילתה למספר משימות, כך שהן תתבצענה במקביל, ולא באופן עוקב כמתואר בדוגמה הקודמת. נתבונן במשימת הסריקה, הדורשת סריקה של שתי טבלאות. במקום לבצע את סריקת שתי הטבלאות באופן עוקב במעבד אחד, ניתן לחלק את משימת הסריקה למספר משימות קטנות יותר - אחת לסריקת טבלת המכירות, ואחרת לסריקת טבלת הלקוחות. נניח, שסריקת כל אחת מהטבלאות אורכת 2 דקות בלבד. בדרך זו מצטמצם משך ביצוע השאילתה ל-12 דקות בלבד.



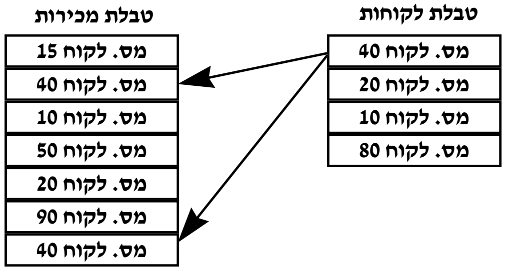
תרשים 8.6: מיקבול של שלב הסריקה בלבד.

ניתן להביא אפילו לרמת מיקבול גבוהה יותר, אם נחלק טבלה אחת למספר מחיצות (Partitions), כך שניתן לעבד מספר מחיצות (בו-זמנית) על ידי מעבדים שונים. נושא חלוקת הטבלה למחיצות מוסבר בפירוט רב יותר בהמשך הפרק. לדוגמה, אם נחלק כל אחת מהטבלאות לשתי מחיצות, כך שכל אחת תכיל 15 מיליון שורות, נוכל להקצות את משימות הסריקה לארבעה מעבדים שונים - 2 מעבדים יסרקו את שתי המחיצות בטבלת המכירות ו-2 מעבדים יסרקו את שתי המחיצות בטבלת הלקוחות. בהנחה שסריקת מחיצה בטבלת המכירות אורכת דקה אחת בלבד, מצטמצם משך זמן הסריקה מ-5 דקות לדקה אחת. בדרך זו, משך ביצוע השאילתה כולה מצטמצם ל-11 דקות. מובן, שחלוקת הטבלאות למספר רב יותר של מחיצות, ושימוש במספר רב יותר של מעבדים, עשויים לגרום לשיפור נוסף בזמן הסריקה.

צירוף מקבילי של טבלאות (Parallel Table Join)

השלב הבא, לאחר סריקת הטבלאות ושליפת השורות הרלוונטיות, הוא ביצוע פעולת צירוף הטבלאות. קיימות מספר שיטות שונות לביצוע פעולה זו, וביניהן Nested Loop Join, Sort Merge Join, Hash Join ועוד. כפי שנראה, חלק מהשיטות ניתנות למיקבול באופן נוח יותר מאחרות.

השיטה הבסיסית לביצוע צירוף שתי טבלאות היא שיטת **הלולאה המקוננת** (Nested Loop). בשיטה זו מתחיל תהליך הצירוף על ידי לולאה חיצונית הסורקת את כל השורות בטבלה אחת. עבור כל שורה מתחילה לולאה מקוננת (פנימית) בה סורקים את הטבלה השנייה, למציאת שורות מתאימות. ניתן לסרוק את הטבלה השנייה באופן סדרתי (מאוד לא יעיל), או תוך שימוש באינדקס, במידה וכזה קיים.



תרשים 8.7: צירוף שתי טבלאות בשיטת לולאה מקוננת.

תרשים 8.7 מדגים את הצירוף בשיטת הלולאות המקוננות. הצירוף מתבצע בין טבלת הלקוחות (מימד) לטבלת המכירות (עובדות). הלולאה הראשית סורקת את טבלת הלקוחות. השורה הראשונה מתייחסת ללקוח שמספרו 40. עבור לקוח זה מתחילה לולאה פנימית, המאתרת את כל שורות המכירות ללקוח זה. בהנחה שבטבלת הלקוחות יש n שורות, ובטבלת המכירות m שורות, שיטה זו תבצע n x m השוואות. בדוגמה הקודמת נמצאו שתי שורות כאלה, עבור לקוח שמספרו 40. שיטה זו מתאימה בעיקר עבור טבלאות קטנות, במיוחד במצב בו שתי הטבלאות נמצאות בזיכרון. היא אינה ניתנת למיקבול בקלות, בשל התלות בין הלולאה הראשית ללולאה הפנימית.

שיפור משך זמן הביצוע של שיטת הלולאה המקוננת יתקבל אם לטבלת המכירות יהיה אינדקס לפי מספר לקוח. במצב זה, תהיה הלולאה הפנימית יעילה מאוד, מכיון שהגישה לשורות המתאימות בטבלת המכירות לא תתבצע על ידי סריקה, אלא על ידי גישה באמצעות האינדקס.

כעת, נציג שיטה אחרת לצירוף טבלאות, שניתנת למיקבול ביתר קלות, שיטת המיון-מיזוג. בשיטה זו, מתחיל צירוף שתי הטבלאות בשלב מיון, וממשיך בשלב שני במיזוג שתי הטבלאות הממוינות. רק שורות שנמצאו בשתי הטבלאות יישלפו לטבלה התוצאתית.

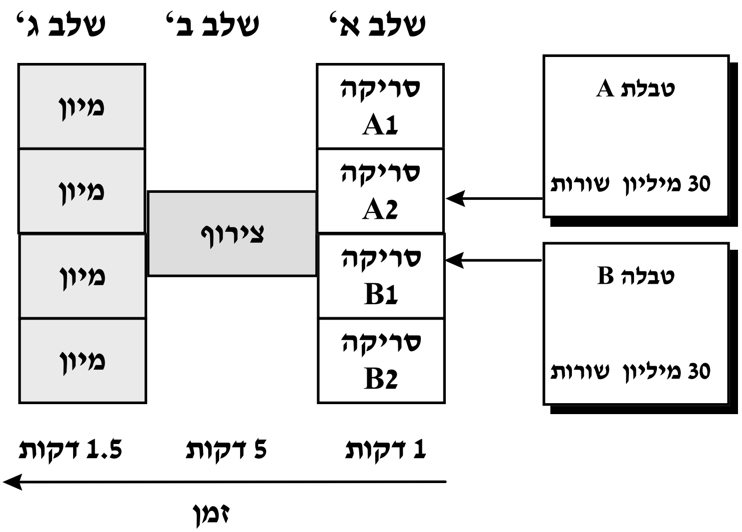


תרשים 8.8: ביצוע צירוף על ידי מיון-מיזוג.

מכיון שתהליך זה מתחיל בשלב מקדים של מיון, ניתן להפעיל תהליך זה במקביל על ידי שני מעבדים שונים, כך שכל מעבד יבצע מיון של טבלה אחת. קיימים אלגוריתמים המאפשרים גם מיזוג מקבילי. הם עושים זאת על ידי חלוקת שתי הטבלאות לטבלאות משנה, מיזוג בנפרד של כל חלק ואחר כך מיזוג התוצאות.

מיון מקבילי (Parallel Table Sort)

בנוסף לשיפור שהושג כתוצאה ממיקבול תהליך הסריקה, ניתן להשיג שיפור נוסף בזמן העיבוד. זאת, על ידי מיקבול השלבים הבאים - הצירוף והמיון. לדוגמה, ניתן לקחת את תוצאת שלב הצירוף ולהקצות את משימת המיון למספר מעבדים. כך, כל מעבד ממיין חלק מהשורות, ולאחר מכן מבצעים מיזוג של השורות הממוינות לקבלת התוצאה הסופית. נניח, שעל ידי מיקבול שלב המיון בסביבה עם ארבעה מעבדים, ניתן להגיע לצימצום משך זמן המיון מ-5 דקות ל-1.5 דקות. בדרך זו משך זמן ביצוע כולל של השאילתה יהיה 7.5 דקות.



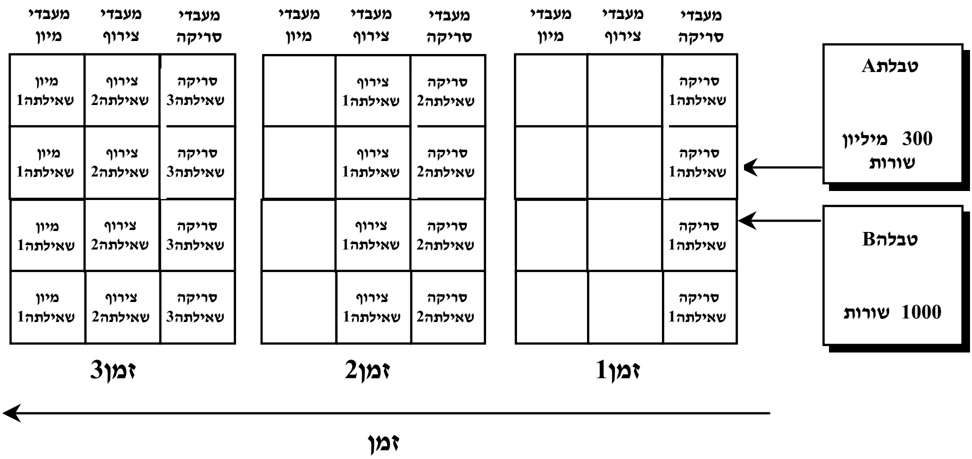
תרשים 8.9: מיקבול שלב הסריקה והמיון.

התמונה המוצגת כאן היא מלאכותית, מפני שהיא מתעלמת מהתקורה. תקורה זו נובעת מהעבודה הנוספת המתבצעת על ידי רכיב האופטימיזציה, כדי לבצע את חלוקת השאילתה לשלבים נפרדים, ומהעבודה הנוספת הנדרשת כדי למזג (Merge) את תוצאות סריקת המחיצות לטבלה אחת לצורך המשך ביצוע השאילתה. עם זאת, התקורה הנוספת נמוכה משמעותית לעומת החיסכון המושג על ידי העיבוד המקבילי של השאילתה. ככל שמספר השורות אותן יש לסרוק גדול יותר, תקורה זו תהיה נמוכה יותר - ביחס לזמן הביצוע הכולל של השאילתה.

זרימה מקבילית (Query Pipelining)

עד כאן התייחסנו לשאילתה בודדת. במציאות אמורה המערכת לטפל בסביבה מרובת משתמשים, ולכן גם סביר להניח שבכל נקודת זמן יהיה צורך לטפל במספר שאילתות. כאן נכנסת לתמונה תפיסת **צינור הזרימה המקבילית** (Pipelining). תפיסה זו אינה מאיצה את משך ביצועה של שאילתה בודדת, אלא את משך ביצוען של אוסף שאילתות. הרעיון הבסיסי הוא לאפשר תחילת ביצוע של שאילתה חדשה, עוד לפני שמסתיים השלב האחרון של השאילתה הנוכחית. כלומר, ליצור מקביליות לא רק **בתוך** השאילתה (Intra Query Parallelism) אלא גם מקביליות **בין** שאילתות (Inter Query Parallelism). שיטה זו מאפשרת למספר רב של מעבדים לעבוד כל אחד על שלב שונה בביצוע שאילתות שונות. אם במערכת רצה שאילתה אחת בלבד, חלק מהמעבדים יהיה בהמתנה, מכיון שיש גבול לרמת המיקבול שניתן ליצור בשאילתה בודדת.

תפיסת צינור הזרימה מאפשרת את ניצול המעבדים לעיבוד שלבים של שאילתות נוספות. נבחן מה היה קורה אם המערכת היתה מורכבת מ-12 מעבדים ומופעלות בה 3 שאילתות זהות. ברגע ש-4 מעבדים היו מסיימים את שלב הסריקה הם היו מעבירים את התוצאה ל-4 מעבדים אחרים, לביצוע הצירוף, והיו מיד מתחילים לבצע את שלב הסריקה לשאילתה הבאה. בינתיים, 4 המעבדים המבצעים את הצירוף היו מסיימים את משימתם ומעבירים את התוצאה ל-4 מעבדים אחרים, לביצוע המיון. בדרך זו, בכל נקודת זמן עסוק מספר רב של מעבדים בביצוע שלבים שונים של שאילתות שונות, או כפי שמקובל לומר - צינור העיבוד מלא כל הזמן במשימות.



תרשים 8.10: זרימה מקבילית של מספר שאילתות על מספר מעבדים.

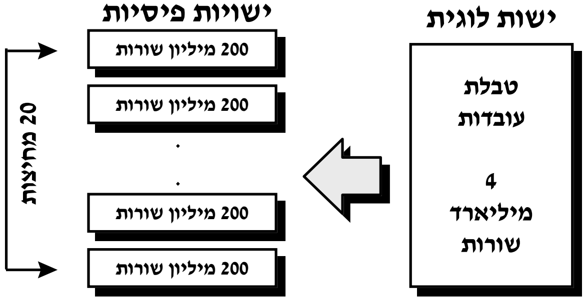
במציאות יכול כל מעבד לבצע כל משימה, ואין מעבדים מיוחדים המבצעים רק משימות סריקה וכאלה המבצעים רק משימות מיון. בתרשים 8.10 ניתן לראות כיצד שאילתה 1 זורמת בצינור העיבוד, ועוברת בשלבים השונים בין מעבדים שונים. למשל, בנקודת זמן 3 יש בצינור 3 שאילתות, כל אחת בשלב עיבוד שונה - שאילתה 1 בסריקה, שאילתה 2 לאחר סריקה ובתוך הצירוף, ושאילתה 3 שכבר סיימה את שלבי הסריקה והצירוף ונמצאת בשלב המיון.

תפיסת צינור הזרימה המקבילית מנצלת בצורה טובה יותר את אוסף המעבדים, על ידי כך שמונעים זמני בטלה עד סיום שאילתה מסוימת. בצורה זו, לאחר פרק זמן ראשוני של מילוי הצינור, כל המעבדים עסוקים כל הזמן.

חלוקת טבלה למחיצות (Table Partitioning)

אחד האתגרים העיקריים בניהול בסיס הנתונים בסביבת מחסן הנתונים היא נפחי הנתונים הגדולים. נפחי נתונים אלה מציבים אתגר מיוחד במספר רב של היבטים - ביצועי השאילתות, טעינת הנתונים ובניית האינדקסים, תהליכי גיבוי ושחזור ועוד. קיים הבדל מהותי בין ניהול טבלה המכילה מספר עשרות, מאות אלפי או אפילו מספר מיליונים של שורות, לבין טבלה המכילה מספר מאות מיליונים או מיליארדים של שורות. לדוגמה, במחסן נתונים של חברת טלקומוניקציה, האוספת מדי יום 50 מיליון רשומות פירוט שיחות, ומנהלת היסטוריה של שנה אחת בלבד, מספר השורות בטבלת העובדות של פירוט השיחות יהיה מעל 4 מיליארד שורות (12 x 30 x 50,000,000). אם לכל שורה נשמור 50 בתים בלבד, נקבל שטבלת העובדות לבדה תופסת נפח של 200 ג'יגה-בית, וזאת ללא אינדקסים ושטחי עבודה. טבלה בגודל כזה תתפרס על פני מספר דיסקים פיסיים שונים, ותהיה קשה מאוד לניהול. אפילו תהליך פשוט כגון גיבוי הטבלה, יכול לארוך שעות רבות, שלא לדבר על תהליכי טעינה ובניית אינדקסים.

מכל האמור לעיל ברור, שניהול פשטני של טבלאות גדולות אינו סביר, וכי נדרש פתרון המותאם לניהול נפחי נתונים גדולים. אחת הטכניקות הנפוצות ביותר לטיפול בבעיה זו היא חלוקת הטבלה ל**מחיצות** (Partitions). הרעיון העומד בבסיס טכניקה זו הוא לקחת יישות לוגית אחת (טבלת העובדות, למשל) ולחלק אותה למספר יישויות פיסיות, יישויות שבסיס הנתונים מכיר ומנהל. את רעיון החלוקה למחיצות קל מאוד להבין, אולם לא כל כך קל ליישם.



תרשים 8.11: חלוקת טבלה למחיצות פיסיות.

בתרשים 8.11 מוצגת טבלה גדולה, המכילה 4 מיליארד שורות, אותה חילקנו למחיצות שוות, כל אחת מכילה 200 מיליון שורות. מבחינה לוגית, זו טבלה אחת. אולם מבחינה פיסית, מוצגות בפנינו 20 מחיצות, כך שכל מחיצה היא יישות ברת ניהול. כלומר, ניתן לגבות אותה בנפרד ממחיצות אחרות, ניתן לבנות לה אינדקס בנפרד מהמחיצות האחרות, ניתן להוסיף מחיצה חדשה לטבלה, או לחילופין ניתן לבטל מחיצה בטבלה.

מדוע לחלק למחיצות

נסקור כמה מהסיבות העיקריות לכדאיות בחלוקת טבלה גדולה למספר מחיצות פיסיות.

ביצועים

חלוקת הטבלה למחיצות מאפשרת בניית טכניקות אופטימיזציה מתוחכמות, לשיפור ביצועי שאילתות SQL. לדוגמה, אם בשאילתה מבקשים את פירוט השיחות לתקופה מסוימת, רכיב האופטימיזציה יכול לזהות איזה מחיצות יש לסרוק ואיזה לא, ובכך לחסוך גישה למיליוני רשומות ואינדקסים. החלוקה למחיצות מתאימה מאוד גם לנושא העיבוד המקבילי. זאת, מכיון שניתן להקצות מעבדים שונים לסריקת מחיצות שונות, במקביל. קיימות מספר ארכיטקטורות חומרה, כמו MPP, הפועלות באופן טבעי עם מחיצות בטבלאות המפוזרות בדיסקים שונים, ולכן תפיסת המחיצות מתאימה להן מאוד.

כושר גידול (Table Scalability)

רעיון חלוקת הטבלה למחיצות מספק יתרון חשוב - יכולת טבעית לגידול הטבלה במשך הזמן. במצב רגיל, בה מנוהלת הטבלה ללא מחיצות, ברור שככל שהטבלה תגדל ילך הקושי בניהולה ויגדל. שיטות העבודה שנבנו לניהול טבלה קטנה אינן מתאימות יותר, ברגע שהטבלה עוברת גודל מסוים. אז, צריך לשנות את כל תפיסת העבודה. לעומת זאת, שימוש במדיניות חלוקת הטבלה למחיצות, כבר בשלבים הראשונים של בניית הטבלה, מספקת מנגנון טבעי לגידול הטבלה, ללא הוספת גורם חדש של סיבוכיות. ממילא כל תפיסות העבודה, הגיבוי והניהול מבוססות על החלוקה למחיצות ולכן, הוספת מחיצה חדשה לטבלה היא טבעית ואינה מזעזעת את כל שיטות העבודה.

ניהול טבלאות גדולות

חלוקת הטבלה למספר מחיצות יכולה לפשט בצורה משמעותית מאוד את בעיות ניהול הטבלה. בדוגמה הקודמת, במקום לנהל טבלה אחת ענקית המכילה 4 מיליארד שורות (שהיא ללא ספק יישות כמעט בלתי אפשרית לניהול), ננהל 20 מחיצות, שכל אחת מהן מכילה 200 מיליון שורות. אם, למשל, נחליט על מדיניות שכל מחיצה מכילה נתוני חודש אחד בלבד, טעינת הנתונים הופכת לפשוטה יותר. את הנתונים החדשים נטען למחיצה חדשה, ובסיום הטעינה נחבר את המחיצה לטבלה. מכיון שכל פעם אנו רק מוסיפים מחיצה חדשה ומוחקים את המחיצה הישנה ביותר, נוצר מצב בו המחיצות אינן משתנות. אי לכך, אין גם צורך לבנות כל פעם מחדש אינדקס לכל הטבלה, אלא רק למחיצה הרלוונטית.

גיבוי ושחזור טבלאות גדולות

מכיון שרוב המחיצות אינן משתנות, אין צורך לגבות אותן מדי תקופה, אלא רק פעם אחת. בעת יצירת או עדכון המחיצה, במידה והיא מתעדכנת. מחיצות ישנות ניתן לסמן כמחיצות לקריאה בלבד (Read Only Partition), כדי להגן עליהן מפני עדכון כלשהו. במקרה של תקלה ניתן לשחזר רק את המחיצה הרלוונטית, ולא את כל הטבלה. למרות שרוב מערכות RDBMS תומכות בגיבוי אינקרמנטלי, עדיין שיטת ניהול המחיצות יעילה מאוד, יחסית לשיטה האינקרמנטלית המחייבת סריקת הטבלה לאיתור הבלוקים בהם חל שינוי.

שיטות לחלוקת טבלה למחיצות

קיימות מספר רב של שיטות לחלוקת טבלה לוגית למספר מחיצות פיסיות. השיטה הנפוצה ביותר היא החלוקה האופקית. בשיטה זו מחלקים את שורות הטבלה למחיצות שונות. בנוסף לשיטה זו קיימות גם שיטות לחלוקה ורטיקלית, בה מחלקים שורות של טבלה אחת בין מספר טבלאות פיסיות שונות. נסקור בקצרה את השיטות העיקריות לחלוקה למחיצות.

הקצאה אופקית למחיצות (Horizontal Table Partitioning)

השיטה הנפוצה ביותר לחלוקת הטבלה למחיצות היא הקצאה אופקית. כלומר, הקצאת חלק משורות הטבלה למחיצה מסוימת, וחלק אחר למחיצה אחרת. למספר השורות המוקצות למחיצה יש השפעה רבה על טיב ביצועי המערכת. כל מחיצה יכולה להמצא בקטעי דיסק שונים, או בדיסקים שונים, כך שמעבדים שונים יכולים לעבד מחיצות שונות במקביל. רכיב האופטימיזציה של בסיס הנתונים מחלק את השאילתה למספר משימות שיכולות לפעול במקביל, ומקצה משימות שונות למעבדים שונים. מכיון שכל מעבד פועל מול מחיצה מסוימת, יש חשיבות בהקצאה מושכלת של שורות הטבלה למחיצות. רצוי שלכל מחיצה יוקצו מספר שווה, פחות או יותר, של שורות. אם מספר השורות המוקצות למחיצה אינו אחיד, נקבל זמני עיבוד ארוכים במעבד אחד, מול זמני המתנה ארוכים במעבד אחר. חוסר איזון זה בזמני עיבוד המחיצות יגרום, בסופו של דבר, לניצול לא טוב של המעבדים ולמצב בו יעילות העיבוד המקבילי נפגמת באופן משמעותי.

הבעיה היא שקל מאוד להגיע לסכמות לא מאוזנות בהקצאת שורות מטבלה למחיצות. לדוגמה, אם נקצה את שורות טבלת נתוני המכירות לפי חודש המכירה, ניתן בקלות להגיע למצב לא מאוזן, בשל ההתנהגות העונתית של המכירות - בחודשי חגים יש עלייה ניכרת במכירות, לעומת חודשים בהם אין חגים. אם נקצה את השורות לפי שנים, יכול המצב להיות אף גרוע יותר, מכיון שסביר להניח שרוב השאילתות פונות בעיקר למחיצה המנהלת את נתוני השנה הנוכחית.

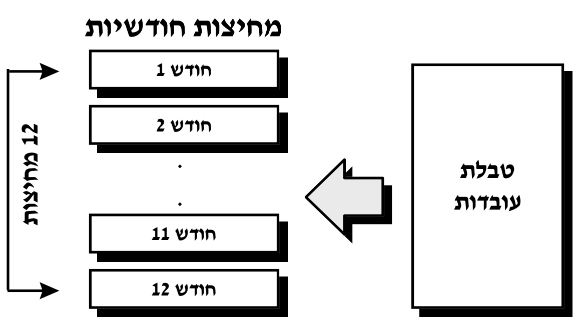
שיטת החלוקה למחיצות נפוצה בעיקר עבור טבלת העובדות. הסיבה לכך היא שטבלה זו נוטה להיות גדולה מאוד, ולכן ההחלטה על חלוקתה למחיצות היא טבעית. יחד עם זאת, לעיתים גם טבלאות מימד יכולות להיות גדולות, ואז בהחלט ניתן לשקול לחלק גם אותן למחיצות. לדוגמה, בחברת טלקומוניקציה או בבנק, מימד הלקוחות יכול להיות גדול מאוד, ולכן רצוי לחלק אותו למחיצות (כדי להנות מכל היתרונות ששיטה זו מספקת).

קיימות מספר שיטות בסיסיות להקצאת השורות למחיצות - הקצאה לפי טווח ערכים של תכונה כלשהי, הקצאה לפי סדר הגעה, הקצאה לפי אלגוריתם כלשהו, הקצאה מעורבת (Hybrid) המשלבת בצורה כלשהי יותר משיטה אחת. נסקור בקצרה כל אחת משיטות ההקצאה האלה.

הקצאה לפי טווח ערכים של תכונה (Range Partitioning)

בשיטה זו משתמשים בעמודה כלשהי, ובטווח הערכים שלה, כבסיס להקצאת שורות הטבלה למחיצות. טווח ערכים מסוים מופנה למחיצה אחת, בעוד טווח ערכים אחר מופנה למחיצה אחרת. ההחלטה על העמודה וטווח הערכים תלויה ביישום, ובאופי השאילתות.

ההקצאה הטבעית ביותר לפי טווח ערכים, היא לפי זמן. למשל, בטבלת עובדות המכילה את סך כל המכירות היומיות של כל סניף לכל מוצר, מופיעה העמודה - תאריך מכירה. עמודה זו יכולה לשמש בסיס לחלוקה למחיצות. למשל, כל המכירות בחודש מסוים ישתייכו למחיצה אחת, בעוד המכירות בחודש אחר ישתייכו למחיצה אחרת. אפשר, כמובן, לבנות מחיצות המבוססות על חלוקת זמן אחרת (למשל מחיצות שבועיות, רבעוניות, חצי שנתיות וכדומה). בדומה, ניתן גם לחלק את טבלת העובדות המכילה את פירוט שיחות הטלפון, את הטבלה המכילה תנועות בחשבון בבנק, את הטבלה המכילה את מכירת כרטיסי הטיסה וכדומה. אם מספר השורות בכל מחיצה שווה, פחות או יותר, תתקבל חלוקה אחידה של הטבלה למחיצות. מכיון שהחלוקה מבוססת על עמודה אחת בלבד - זמן - מקובל לקרוא לשיטה זו בשם Single Attribute Partitioning.



תרשים 8.12: חלוקה אופקית לפי עמודה אחת.

כאמור, אסטרטגיית חלוקת טבלה על בסיס זמן היא אסטרטגיה מקובלת מאוד. אסטרטגיה זו מאפשרת לנו לקבוע לטבלה מגבלת גודל, על ידי הגדרת מספר המחיצות שננהל. לדוגמה, אם נחליט לנהל עומק היסטורי של שנתיים, ונחלק את הטבלה לפי רבעונים, נקבל סה"כ 8 מחיצות מתגלגלות. הוספה של רבעון חדש גורמת להורדה מקבילה של מחיצת הרבעון הישן ביותר, יחד עם האינדקס שלה. בצורה זו ניהול העומק ההיסטורי פשוט ויעיל, ואינו דורש שימוש בפקודת SQL לביטול שורות מהטבלה. פקודה זו יקרה מבחינת משאבי המחשב, מכיון שהיא מעדכנת גם את האינדקסים.

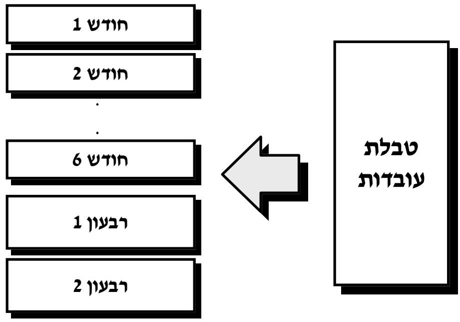
ניתן לבנות שיטת הקצאה המבוססת על מספר עמודות, ולא רק לפי עמודה אחת. למשל, ניתן לחלק את טבלת העובדות המכילה את סך כל המכירות של כל סניף לכל מוצר לכל יום, לפי תאריך ולפי סניף. כלומר, כל המכירות בחודש מסוים ובסניף מסוים, ישתייכו למחיצה אחת והמכירות באותו חודש אך בסניף אחר ישתייכו למחיצה אחרת. אם לרשת השיווק יש 30 סניפים, ונרצה לנהל מחיצה לכל סניף לכל רבעון, יתקבלו סך הכל 60 מחיצות בטבלת העובדות. לשיטה זו מקובל לקרוא Multi Attribute Partitioning.



תרשים 8.13: חלוקה למחיצות לפי שתי עמודות.

בתרשים 8.13 מופיעה דוגמה לטבלת עובדות, המחולקת למחיצות לפי שתי עמודות - רבעונים וסניפים. במידת הצורך ניתן לבצע חלוקה לפי עמודות נוספות, אם כי מצב זה די נדיר. נשים לב שחלוקה למחיצות לפי עמודה שיכולה להשתנות לאורך זמן, היא מסוכנת. בדוגמה הקודמת ארגון מחדש של מערך הסניפים, על ידי איחוד מספר סניפים וביטול סניפים, עשוי לדרוש ארגון מחדש של כל המחיצות, דבר שיכול להיות מורכב מאוד. מכאן ברורה ההמלצה לחלק למחיצות רק לפי עמודות שההסתברות שלהן להשתנות קטנה מאוד.

עד כאן חילקו שיטות החלוקה למחיצות את השורות באופן אחיד בין המחיצות - לפי חודשים, רבעונים וכדומה. לעיתים, קיימים מצבים בהם הגישה לנתונים ישנים נדירה, וכדי להקטין את מספר המחיצות המנוהלות מקצים את השורות באופן לא אחיד. לדוגמה, ניתן לבנות 6 מחיצות חודשיות, עבור חצי השנה האחרונה, ועוד שתי מחיצות לרבעון. בסך הכל נקבל עומק היסטורי של שנה, המחולק באופן לא אחיד בין 8 מחיצות, במקום 12 מחיצות, אם היינו מחלקים באופן אחיד לפי חודשים.



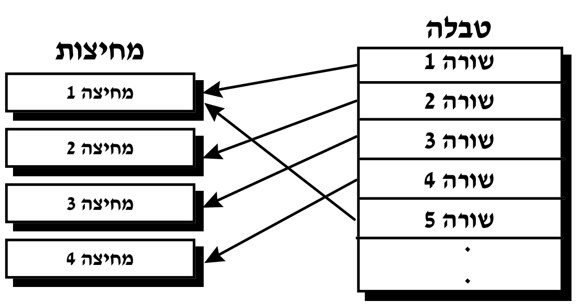
תרשים 8.14: חלוקה לא אחידה למחיצות.

שיטה זו מורכבת מאוד לניהול, מכיון שגלגול המחיצות הוא מורכב. ברגע שמצטרפת מחיצה עבור חודש חדש ברבעון חדש, מתחיל תהליך גלגול מורכב למדי. מחיצת הרבעון השני תבוטל, ובמקומה תבנה מחיצה רבעונית חדשה, המורכבת מהשורות השייכות לשלושת החודשים האחרונים. ניהול מורכב זה מבטל לעיתים את היתרונות שבהקצאה הלא אחידה. לכן, שיטה זו אינה מומלצת לשימוש, אלא במצבים מאוד מיוחדים.

הקצאה על בסיס זמן מתאימה בעיקר לטבלת העבודות. בטבלאות מימד ברור שהבסיס להקצאה אינו יכול להיות הזמן, מכיון שהוא אינו תכונה של המימד, אלא בדרך כלל רק של טבלת העובדות. במקרים אלה ניתן למצוא עמודה אחרת שתשמש בסיס להקצאה למחיצות - למשל, הפריסה הגיאוגרפית לפי מחוזות, או לפי סוגי לקוחות. יש להשתדל ולמצוא עמודה של המימד אשר מחלקת באופן שווה ואחיד, ככל שניתן, את השורות על פני המחיצות. אחרת, יתקבל מצב של חוסר איזון בין מספר השורות למחיצה.

הקצאה לפי סדר הגעה (Round Robin Partitioning)

זו שיטת הקצאה פשוטה - אם קיימות n מחיצות, מקצים את השורה הראשונה למחיצה הראשונה, את השנייה למחיצה השנייה וכך הלאה עד לשורה n, אותה מקצים למחיצה n. השורה הבאה מוקצית למחיצה הבאה, וחוזר חלילה עבור כל הטבלה.

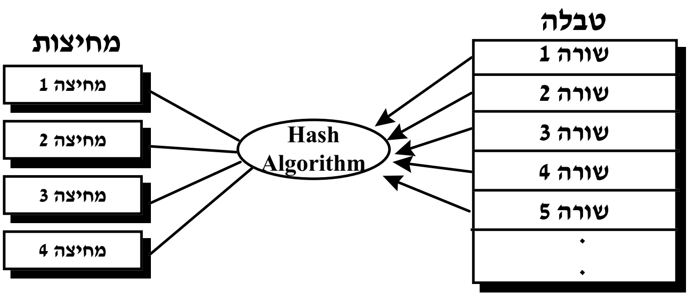


תרשים 8.15: הקצאת שורות למחיצות לפי סדר ההגעה.

לדוגמה, אם אנו מעריכים שבטבלת העובדות יהיו כ-500 מיליון שורות, ואם נרצה לנהל 50 מחיצות, משמעות הדבר היא שגודל כל מחיצה יהיה 10 מיליון שורות. בשיטה זו מספר השורות למחיצה יהיה כמעט זהה, ולכן נקבל איזון טוב בעיבוד המחיצות. החסרון העיקרי של שיטה זו נובע מכך שאין הגיון מסוים מאחורי ההקצאה למחיצות. לכן, בסיס הנתונים אינו מסוגל לנצל הקצאה זו לקבלת החלטה באיזה מחיצות יבוצע החיפוש.

הקצאה לפי אלגוריתם (Hash Partitioning)

ניתן לראות בשיטת הקצאה זו שיטה מופשטת יותר של שיטת ההקצאה לפי טווח. בשיטה זו מופעל אלגוריתם כלשהו (Hash Algorithm), המבצע חישוב על עמודה כלשהי בטבלה, ועל פי תוצאת האלגוריתם מתקבלת ההחלטה לאיזה מחיצה להקצות את השורה.



תרשים 8.16: הקצאת שורות למחיצות לפי אלגוריתם.

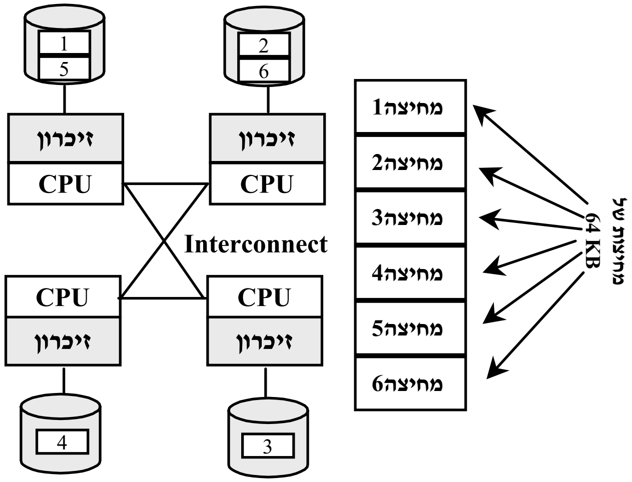
טיב פיזור השורות בין המחיצות תלוי במידה רבה בטיב האלגוריתם, ובאופי ערכי העמודה. גם שיטה זו, בדומה לשיטת ההקצאה לפי סדר הגעה, אינה תומכת בחיפוש שורות לפי טווח ערכים, מכיון שהשורות הרלוונטיות יכולות להיות מפוזרות בין מספר רב של מחיצות.

חלוקה למחיצות בשיטה ורטיקלית (Vertical Partitioning)

בשיטה זו, החלוקה למחיצות אינה מתבצעת לפי שורות הטבלה, אלא לפי עמודותיה. אם קיימת קבוצת משתמשים אחת המתעניינת בעיקר בעמודות מסוימות, בעוד קבוצה אחרת מתעניינת בעמודות אחרות, ניתן לפצל את הטבלה לשתי טבלאות נפרדות. כך, תכיל כל טבלה רק את העמודות הרלוונטיות. כמובן, שיש לשכפל את המפתח העיקרי של הטבלה לכל אחת מהטבלאות החדשות, דבר שעלול להגדיל את נפח הדיסק הנדרש. למרות הגידול בנפח הדיסק הנדרש לאחסון הטבלאות המפוצלות, ניתן להגיע לשיפור בביצועי השאילתות. זאת, מכיון שכל טבלה מכילה פחות עמודות, ולכן ניהולה בדיסק יעיל יותר ושטחי הזיכרון הדרושים לעיבודה קטנים יותר.

חלוקה למחיצות מבוססת חומרה (Data Striping)

בארכיטקטורות MPP קיימים מנגנונים המבצעים חלוקה למחיצות בחומרה עצמה. מטרת פיזור זה של הנתונים בין הדיסקים השונים היא לאפשר למשימות מקבילות לפעול באופן בלתי תלוי בכל צומת של מערכת MPP, ללא צורך בהעברה מתמדת של נתונים בין הצמתים. המנגנון מבצע חלוקה של טבלה גדולה למספר מחיצות קטנות יותר, ומפזר אותן בין הדיסקים השונים. גודל המחיצה נקבע על ידי פרמטרים שונים של החומרה, ובאופן שיאפשר סריקה מקבילה יעילה בטבלה. לדוגמה, אם נקח טבלת עובדות בגודל 10GB, ונפזר אותה על פני 10 צמתים במערכת MPP (על בסיס פרמטר חלוקה של 64KB), יתקבל אוסף מחיצות בגודל 64KB, המפוזרות באופן שווה ואחיד בין עשרת הדיסקים. סריקה של טבלה זו יכולה להתבצע על ידי 10 מעבדים, הפועלים במקביל במחיצות הנמצאות באחריותם. פיזור אחיד זה של מחיצות מקטין למינימום את הצורך בהעברת הנתונים ובהעמסת ערוץ Interconnect.



תרשים 8.17: פיזור מחיצות בארכיטקטורת MPP.

תרשים 8.17 מדגים חלוקה ופיזור על ידי החומרה בארכיטקטורת MPP. הטבלה מחולקת למחציות בגודל 64KB, ומפוזרת באופן אוטומטי על ידי החומרה. נשים לב לכך שהחלוקה מתבצעת על ידי פיזור יחידות בגודל 64KB, בכיוון השעון. לאחר שמסתיים סיבוב החלוקה הראשון, של 4 מחיצות, ממשיך הסיבוב הבא, וכך עד לסוף הטבלה. מכיון שהפיזור מתבצע ברמת החומרה, או מערכת ההפעלה, בסיס הנתונים אינו יודע היכן נמצאת כל מחיצה. שיטה זו טובה אם הסריקות מתפזרות באופן אחיד על פני כל המחיצות. אם רוב הסריקות ממוקדות במחיצות מסוימות בלבד, הצמתים המכילים מחיצות אלה יהוו, קרוב לוודאי, צוואר בקבוק. לכן, חלק מעוצמת המערכת המקבילית לא תנוצל.

מהי השיטה המומלצת לחלוקה למחיצות

ההחלטה לגבי שיטת החלוקה למחיצות תלויה, במידה רבה, בנפח הנתונים ובפרופיל השאילתות הטיפוסיות, אך גם בארכיטקטורת החומרה. אם, למשל, אנו רואים ש-90% מהמשתמשים מתעניינים בעיקר במכירות הסניף שלהם במספר שבועות אחרונים, החלוקה למחיצות לפי חנויות וזמן תהיה מאוד יעילה, מכיון שרוב השאילתות יסרקו רק את שורותיה של מחיצה מסוימת. הפעלת שאילתות מסוג זה, מול טבלה שאינה מחולקת למחיצות תהיה הרבה יותר איטית, מכיון שהיא סורקת מספר שורות גדול בהרבה.

קביעת טכניקת החלוקה הנכונה בסביבה עתירת שאילתות אד הוק, היא כמובן משימה מורכבת מאוד. הסיבה לכך היא שטכניקת חלוקה אחת תהיה טובה לסוג מסוים של שאילתות, וגרועה מאוד לסוג אחר של שאילתות. לדוגמה, חלוקה למחיצות לפי סניפים תהיה מאוד לא יעילה לשאילתות המתייחסות למכירות מוצרים מסוימים, מכיון שמכירות אלה יכולות להופיע בכל המחיצות.

תמיכת בסיס הנתונים בחלוקה למחיצות

רוב ספקי מערכות RDBMS שילבו את התמיכה בניהול מחיצות אופקיות כחלק אינטגרלי מהמערכת. תמיכה זו מאפשרת למנוע מבסיס הנתונים להגדיר באופן אוטומטי את שיוך שורה למחיצה מסוימת, ולהשתמש בהגדרה זו בעת ביצוע האופטימיזציה של פקודות SQL. בזמן ההחלטה על סריקת טבלאות (Table Scan), מזהה רכיב האופטימיזציה (לפי טווח התאריכים המופיע בשאילתה) איזה מחיצות עליו לסרוק, ואיזה לא. לדוגמה, אם השאילתה מבקשת להציג מכירות חודש מסוים, מול החודש המקביל בשנה קודמת, ניתן לחסוך סריקות במספר רב של שורות אם רכיב האופטימיזציה יתמקד רק בשתי המחיצות הרלוונטיות. גם במקרה של איתור שורה באמצעות אינדקס, יכול רכיב האופטימיזציה להחליט באיזה אינדקס מחיצה להשתמש. לתמיכת בסיס הנתונים בחלוקה למחיצות, יש משמעות (מבחינת זמני התגובה של השאילתות), ותכונה זו הפכה בשנים האחרונות לאחת התכונות החשובות של מערכות RDBMS הפועלות בסביבת מחסן הנתונים.

אם מערכת RDBMS אינה תומכת באופן ישיר בחלוקה למחיצות צריך מנהל בסיס הנתונים (DBA) לבנות תהליך המחלק את השורות לטבלאות שונות, ולבנות מבט לוגי (View) המבוסס על איחוד (Union) כל טבלאות אלו לטבלה לוגית אחת. במקרה זה, רכיב האופטימיזציה אינו מכיר את החלוקה למחיצות, וכמובן שאינו משתמש בו לאופטימיזציה של פקודות SQL. נדגים את רעיון החלוקה למחיצות באמצעות טבלת המכירות. נניח שהחלטנו לחלק את טבלת המכירות לשתי מחיצות בלבד, כך שכל מחיצה תכיל את המכירות של חצי שנה. לכל מחיצה ננהל טבלה נפרדת - Sales_98H1 ו-Sales_98H2. נגדיר מבט לוגי המאחד שתי טבלאות אלה באמצעות פקודת View הבאה:

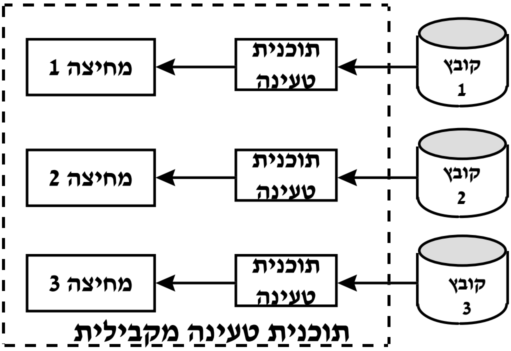
```
1. CREATE VIEW SALES_98 AS
2.    SELECT *
3.    FROM    SALES_98H1
4.    UNION
5.    SELECT *
6.    FROM    SALES_98H2
```

ביצועי שאילתות הפועלות במבט לוגי מסוג זה, תלויים באופן משמעותי בטיב רכיב האופטימיזציה וצורת הטיפול באיחוד טבלאות. פעולת האיחוד היא פעולה יקרה, מבחינת משאבי המחשב. אם רכיב האופטימיזציה יודע לנצל את מקביליות החומרה ניתן לשפר מהותית את הביצועים על ידי פיצול השאילתה למספר תת שאילתות הפועלות במקביל במעבדים שונים, ולבסוף מתבצעת פעולת איחוד טבלאות התוצאה. אפשרות אחרת, ומאוד לא מומלצת, לשיפור הביצועים היא לאפשר למשתמש להכיר את מבנה המחיצות ולהשתמש בשאילתות, באופן מפורש, בשמות הטבלאות הפיסיות. במקרה זה, כל שינוי במבנה המחיצות מחייב שינוי מקביל של השאילתות, כדי לשקף את השינוי. מכיון שכיום רוב מערכות RDBMS תומכות באופן ישיר בחלוקה אוטומטית למחיצות, בעיית הטיפול במחיצות בשאילתות הפך לשקוף, מבחינת המשתמש.

התמיכה בחלוקה למחיצות ורטיקליות חלשה יותר ברוב מערכות RDBMS, בעיקר מפני שהשיטה פחות נפוצה. דוגמה אחת יוצאת דופן היא מערכת Sybase IQ, המנהלת את הטבלה לפי עמודות. עבור שאילתות הממוקדות במספר קטן של עמודות, טכניקה זו יעילה ביותר ויכולה להביא לשיפור דרמטי בזמני התגובה.

טעינה מקבילית (Parallel Load)

תהליך טעינת הנתונים בסביבת מחסן הנתונים הוא אחד התהליכים החשובים ביותר, מכיון שזו הדרך בה מתעדכן מחסן הנתונים. אין זה נדיר לטעון מיליונים רבים של שורות במהלך טעינה אחד. מאחר וחלון הזמן העומד לרשות מהלכי הטעינה הוא מוגבל, יש חשיבות רבה בהקטנת משך הטעינה ככל הניתן. יצרני מערכות RDBMS פיתחו תוכניות שירות מיוחדות, המסוגלות לנצל את ריבוי המעבדים ולבצע תהליכי טעינה מקבילים. בדרך כלל, יש להכין את הנתונים לטעינה בקבצים נפרדים, קובץ נפרד עבור כל מחיצה שיש לטעון. תוכנית הטעינה יכולה להפעיל מספר משימות טעינה במקביל, כך שכל מחיצה נטענת באופן בלתי תלוי במחיצות אחרות. תוכניות אלו טוענות את המחיצות באופן ישיר (Direct Load), ולא על ידי שימוש בפקודות Insert, ולכן הן מהירות מאוד. יחד עם זאת, יש לשים לב שמאחר והן טוענות את הנתונים באופן ישיר, לא מתבצעות בדיקות שלמות ואמינות מול טבלאות אחרות (Integrity Checks). בגמר הטעינה יהיה צורך להפעיל תוכניות מיוחדות לבדיקת אמינות הנתונים.



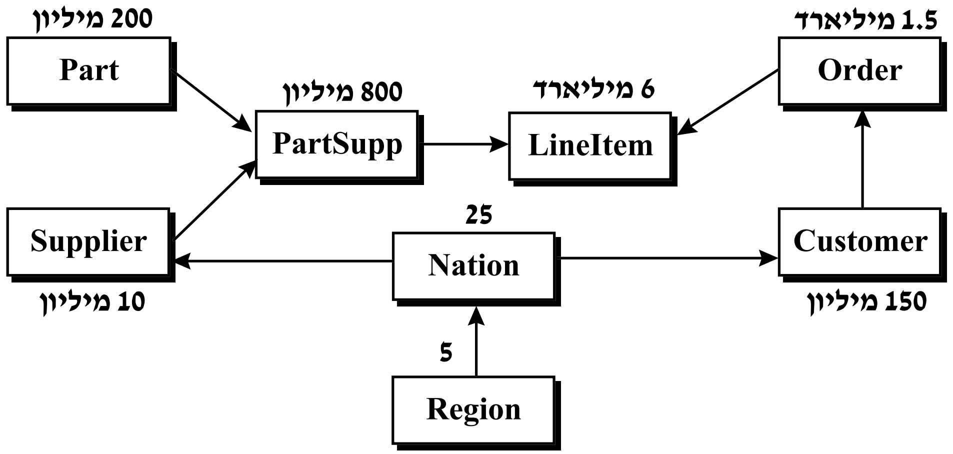
תרשים 8.18: טעינה מקבילית של מחיצות.

מבחני ביצועים לסביבת מחסן נתונים

בעולם מערכות ניהול בסיסי הנתונים מקובל לבחון את ביצועי בסיס הנתונים באמצעות מבחנים שונים. כדי לקבוע מבחן בלתי תלוי ביצרן בסיס הנתונים הוקם ארגון התנדבותי בשם Transaction Processing Council, או בשמו הידוע יותר - TPC. ארגון זה בנה אוסף של מבחני ביצועים לבסיסי נתונים הפועלים בסביבות שונות. עד להופעת מחסני הנתונים, היה המבחן הנפוץ ביותר מבחן TPC-C. מבחן זה מיועד בעיקר לסביבות תפעוליות מקוונות עתירות תנועות, אך אינו מתאים לבחינת ביצועי בסיס הנתונים בסביבת מחסן הנתונים. לכן, לפני מספר שנים פרסם הארגון מבחן חדש, TPC-D, שנועד לבחון את בסיס הנתונים, בעיקר מבחינת עוצמת האיחזור בנפחי נתונים גדולים מאוד (כלומר, בסביבה אופיינית למחסני נתונים), ותמיכה בתהליכי קבלת החלטות.

המבחן מגדיר את מבנה הסכמה, את מספר הרשומות בכל טבלה וכן אוסף של 17 שאילתות שונות, משאילתות פשוטות ועד שאילתות מורכבות מאוד. חלק מהשאילתות מחייבות מספר סריקות של הטבלאות וביצוע צירוף בין מספר טבלאות.

כפי שניתן לראות מתרשים 8.19, הסכמה של מבחן TPC-Dמתייחסת להזמנות מלקוחות. סך הכל יש 150 מיליון לקוחות. לכל הזמנה יש כותרת ושורות. סך הכל יש 1.5 מיליארד כותרות ו-6 מיליארד שורות בהזמנה. כל שורה בהזמנה מתייחסת לפריט ולספק מסוים. יש 200 מיליון חלקים ו-10 מיליון ספקים שונים. גם הלקוחות וגם הספקים שייכים ל-25 אזורים, שמצידם שייכים ל-5 מחוזות.



תרשים 8.19: סכמה למבחן TPC-D.

יצרנים המבקשים לפרסם את תוצאות בחינת TPC-D של המערכות שלהם, חייבים לבצע את המבחן במחשבים מסחריים רגילים, לא במחשבים מיוחדים שעברו שינויים והתאמות לקבלת ביצועים משופרים. יצרן המפרסם את תוצאות המבחן כפי שנערך בדגם חדש, חייב להפיץ את הדגם ללקוחות תוך חצי שנה, לכל המאוחר, מיום פרסום התוצאות. תוצאות המבחן כוללות את תצורת המחשב (מספר מעבדים, זיכרון, נפח דיסקים, שם מערכת ההפעלה וגרסתה), את שם וגרסת בסיס הנתונים שהשתתף במבחן, את זמן טעינת הנתונים, את נפח הנתונים והאינדקסים בדיסק, את ביצועי 17 השאילתות השונות והעלות הכוללת של תצורת המחשב שהשתתף במבחן (עלות כוללת מחושבת על פי עלות רכישת המערכת החדשה ועוד 5 שנות תחזוקה). כל יצרן חייב לפרסם את יחסי עלות/ביצועים (Price/Performance) לכל מערכת.

בשל נפחי הנתונים הגדולים של המבחן והאתגר בביצוע השאילתות, כל היצרנים עושים שימוש מירבי בעיבוד המקבילי בחומרה ומנצלים את היכולת לחלק את הטבלאות למספר מחיצות, לשמור קטעי טבלאות בזיכרון וכדומה.

סיכום

הטכנולוגיה המקבילית היא כיום טכנולוגיית החומרה החשובה ביותר בהקשר למחסני הנתונים. טכנולוגיה זו, על פניה הרבים, על ההתפתחויות הבלתי פוסקות בה, על שימושיה כמעט בכל אחד מתחומי מחסן הנתונים - ניהול בסיס הנתונים, טעינת הנתונים, גזירת נתונים למרכולי נתונים וכדומה - משמשת כתשתית חשובה ליכולת מחסן הנתונים לגדול מבחינת נפח הנתונים, מבחינת מספר המשתמשים שהוא משרת ומבחינת מורכבות היישומים המופעלים בו. כושר הגידול (Scalability) של מחסן הנתונים, שהוא אחד השיקולים המרכזיים של מעצבי מחסני נתונים, נשען במידה רבה על הטכנולוגיה המקבילית, ועל השיפורים הבלתי פוסקים בה.

פרק 9: סקירת התהליכים העיקריים במחסן הנתונים

מבוא

מטרת פרק זה היא לסקור את התהליכים העיקריים המתרחשים בסביבת מחסן הנתונים. כאשר המשתמש מתבונן במחסן הנתונים הוא רואה, בעיקר, את כלי השאילתות שעומדים לרשותו ודרכם את בסיס הנתונים המשרת אותם. זו נקודת מבט פשטנית ומטעה. למעשה, זהו רק קצה הקרחון, כאשר מספר רב של תהליכים מתרחשים מתחת לפני המים, או כפי שמקובל לעיתים לקרוא להם **תהליכי החדר האחורי** (Back Office) של מחסן הנתונים. כפי שנראה, מחסן הנתונים עתיר בתהליכים, חלקם מורכבים למדי.

סיווג התהליכים במחסן הנתונים

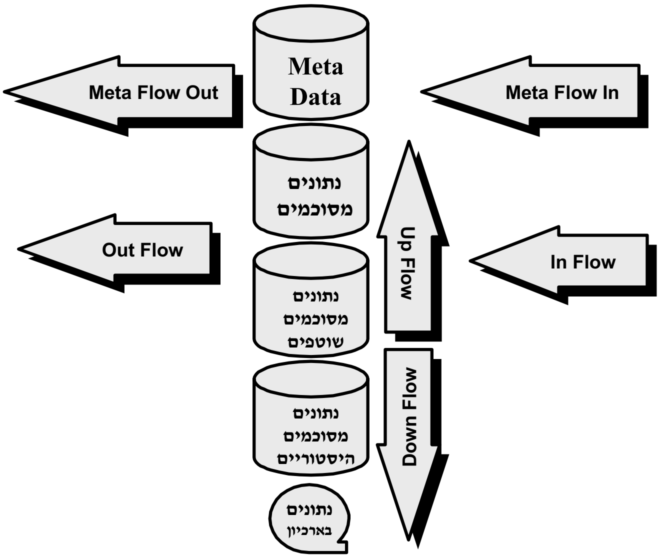
סביבת מחסן הנתונים עתירת תהליכים שונים, העוסקים בהיבטים שונים של ניהול ותפעול מחסן הנתונים. מקובל לסווג את מכלול התהליכים האלה לארבע קטגוריות עיקריות:

- **תהליכי זרימה פנימה** (Inflow Process) - תהליך זה עוסק בזרימה השוטפת של נתונים מהמערכות התפעוליות או החיצוניות פנימה, אל מחסן הנתונים. זהו התהליך המורכב ביותר, והוא עוסק בכל השלבים. ביניהם ניתן למנות את גזירת הנתונים מהמערכות התפעוליות, הבאת הנתונים אל סביבת מחסן הנתונים, מיטוב ושיפור הנתונים, שינוי מבנה הנתונים (למבנה הנדרש על ידי מחסן הנתונים), הכנת טבלאות ביניים לקראת טעינת הנתונים וכלה בטעינת הנתונים אל מחסן הנתונים.
- **תהליכי זרימה מעלה** (Upflow Process) - תהליכים אלה עוסקים בבניית סיכומים (Aggregates) שונים, הנדרשים לקבלת זמני תגובה משופרים לשאילתות. ברמה הבסיסית ביותר נמצאים **נתונים מפורטים שוטפים** (Current Detail Data). מרמת פירוט זו מקובל לבנות רמות שונות של סיכומים. לדוגמה, ניתן לנהל סיכומים שבועיים, חודשיים, רבעוניים ושנתיים. כל רמה כזו מייצגת רמה הולכת וגוברת של הסיכומים. החל ב**סיכום קל** (Light Summarized) ועד לרמות הסך הכללי (Highly Summarized).

❖ **תהליכי זרימה מטה** (Down Flow Process) - תהליכים אלה עוסקים, בעיקר, בטיפול בנתונים המתיישנים. מכיון שמחסן הנתונים צובר כל הזמן נתונים, יש להחליט מהו העומק ההיסטורי שיש לנהל במקוון. נתונים החורגים מחלון זמן זה (למשל, נתוני מכירות מלפני 18 חודשים), מטופלים בצורה כלשהי. קיימות אפשרויות שונות, החל בהעברת הנתונים לטבלאות היסטוריות מיוחדות, העברתן למדיה מגנטית משנית (כגון, קלטות) וכלה בהחלטה על מחיקתם.

❖ **תהליכי הזרימה החוצה** (Out Flow Process) - תהליכים אלה עוסקים בזרימת הנתונים ממחסן הנתונים אל המשתמשים. מעבר לכלי השאילתות הרגילות קיימים מצבים בהם יש להפיץ נתונים שונים לצרכנים שונים, להצביע על חריגים ולהפעיל מנגנוני התראות. בנוסף, יש להעביר נתונים מהמחסן הארגוני אל מרכולי הנתונים.

❖ **תהליכי זרימת מידע על הנתונים** (Meta Data Flow Process) - תהליכים אלה עוסקים בזרימת המידע המאוחסן במילון הנתונים (Repository), שעיקרו מידע על הנתונים שבמחסן הנתונים. מידע זה נע אל מילון הנתונים וממנו החוצה, כאשר דרוש. מילון הנתונים מכיל מידע רב לגבי סוגי הנתונים שבמחסן הנתונים, תאריך עדכונם האחרון, נוסחאות הגזירה, מבנה הנתונים וכדומה.



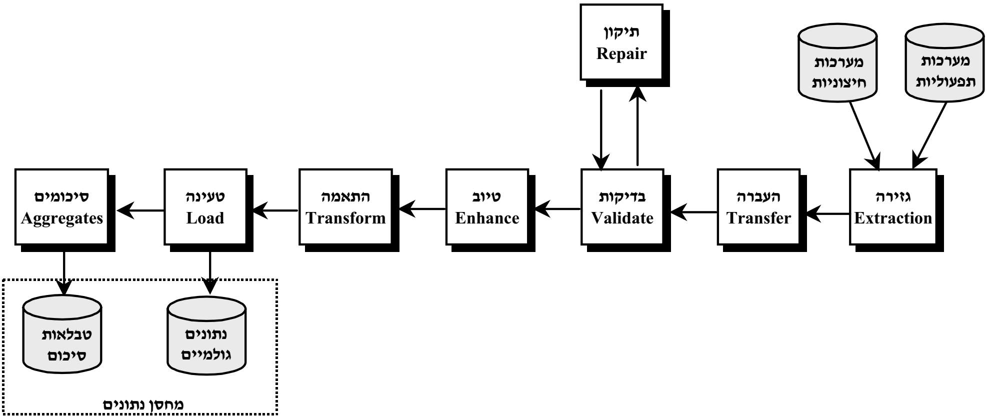
תרשים 9.1: תהליכי הזרימה העיקריים בסביבת מחסן הנתונים.

סקירת תהליכי זרימה פנימה (In Flow Process)

מבט-על של תהליכי הזרימה פנימה

תהליכי הזרימה פנימה עוסקים בגזירת הנתונים ממקורותיהם (מערכות תפעוליות או מערכות חיצוניות), ניקוי הנתונים, שיפור וביצוע המרות שונות (הן בתוכן והן במבנה הנתונים) ועד טעינתם אל מחסן הנתונים. לתהליכים אלה מקובל גם לקרוא Data Extraction, Enhancement and Transformation.

באופן כוללני יותר ניתן לומר שתהליכים אלה עוסקים במקרה פרטי של נושא רחב יותר: העברת נתונים (Data Movement) בין סביבות מחשוב שונות. לא מקרה הוא שיצרני כלי העברת והסבת נתונים (Data Migration) נכנסו לתחום זה, והתאימו את המוצרים שלהם לדרישות הייחודיות שהציבה סביבת מחסן הנתונים.



תרשים 9.2: שלבים עיקריים בתהליך הזרימה פנימה.

נפרט בקצרה כל אחד מהשלבים המתבצעים בתהליך זרימת נתונים פנימה, אל מחסן הנתונים:

❖ **גזירת הנתונים** (Data Extraction) - בשלב הזה מתבצעת גזירת הנתונים הרלוונטיים מתוך המקורות. בדרך כלל בפעם הראשונה מתבצעת גזירה מלאה (Full Propagation) ובהמשך גזירות אינקרמנטליות של השינויים בלבד (Changed Data Propagation). ברור שגזירה מלאה הנה פשוטה יחסית בעוד שגזירה של שינויים בלבד יכולה להיות מורכבת יותר בגלל הצורך בזיהוי השינויים שבוצעו מאז הגזירה האחרונה. קיימות טכניקות שונות של גזירת שינויים.

- **העברת הנתונים** (Data Transfer) - הסיום של תהליכי הגזירה הוא אוסף של קבצי ביניים הממתינים להעברה לסביבת מחסן הנתונים. העברת קבצים אלה לסביבת מחסן הנתונים יכולה להתבצע במיגוון רחב של אפשרויות - שימוש בתוכנת File Transfer כלשהי, שימוש בכלי ביניים מיוחדים (Middleware Gateways) המאפשרים גישה ישירה מסביבת מחסן הנתונים אל פלטפורמות מחשוב אחרות למשיכת הקבצים כדומה.

- **בדיקת נתונים** (Data Validating and Cleaning) - בשלב זה מתבצע תהליך בדיקת איכות וניקוי הנתונים. הנתונים נבדקים (מבחינת תקינותם) מול טבלאות שונות, מתבצעות בדיקות טווחים, ולעיתים בדיקות מורכבות יותר, לאיתור וזיהוי מצבים לא תקינים בנתונים. במידת האפשר מתבצעים תהליכי תיקון בנתונים, אם כי הדרך הטובה ביותר היא לבצע את התיקונים במערכות המקור. מכיון שלא תמיד ניתן לבצע את התיקונים במערכות המקור, ניתן לבצע את התיקונים גם בשלב זה.

- **טיוב הנתונים** (Data Enhancement) - בשלב זה מתבצע תהליך שיפור איכות הנתונים, על ידי הפעלת בדיקות תקינות שונות על תוכן הנתונים. נתונים נבדקים מול טבלאות לתקינות ערכים וטווחים, ערכים חסרים מושלמים, קודים מתורגמים לקודים אחידים, פורמטים שונים מומרים לפורמט אחיד, סתירות בנתונים נפתרות (למשל, אותו לקוח מופיע בשתי כתובות שונות בשני מקורות שונים), שדות מחולקים לתת שדות (כמו למשל כתובת מפורקת לרחוב, מספר, עיר) וכדומה.

- **שינוי מבנה הנתונים** (Data Transforming) - מכיון שמבנה הנתונים במחסן הנתונים אינו זהה למבנה הנתונים כפי שבא לידי ביטוי במקורות, יש לבצע התאמה במבנה הנתונים. לעיתים, ההתאמה פשוטה מאוד, אולם היא יכולה להיות גם מורכבת מאוד. לדוגמה, שני קבצי מקור הופכים לקובץ יעד אחד, קובץ מקור אחד הופך למספר קבצי יעד, מפתחות טבעיים הופכים למפתחות מלאכותיים, קודים מומרים לקוד אחיד וכדומה.

- **טעינת הנתונים** (Data Load) - בשלב זה כל הנתונים מוכנים לקראת טעינה למחסן הנתונים. בדרך כלל מתבצעת הטעינה באמצעות תוכניות שירות מיוחדות, המבצעות טעינה מהירה תוך ניצול ריבוי המעבדים בחומרה. לאחר סיום טעינת הנתונים יש לבנות מחדש את האינדקסים עבור הטבלאות שעודכנו.

- **הכנת הסיכומים** (Aggregates Generation) - עם סיום טעינת הנתונים ניתן להכין את כל טבלאות הסיכום. ניתן לראות את תהליכי הכנת הסיכומים כסיום תהליכי הזרימה פנימה, או כחלק מתהליכי הזרימה מעלה. נפרט את הכנת טבלאות הסיכומים בסעיף נפרד, העוסק בתהליכי הזרימה מעלה.

תהליכים אלה הם המורכבים ביותר להגדרה ובנייה ודורשים חלק ניכר מהמשאבים הכרוכים בבניית מחסן הנתונים. ככל שמקורות הנתונים מגוונים יותר, איכות הנתונים במערכות התפעוליות נמוך יותר ומבנה הנתונים רחוק מהמבנה הנדרש במחסן הנתונים - כך גבוה יותר המאמץ הנדרש בשלב זה. מסטטיסטיקות שונות שנאספו על ידי ארגונים שיישמו את תפיסת מחסן הנתונים מתברר שההשקעה בשלב זה יכולה להגיע עד כדי 70%-80% מהמאמץ הכולל.

ארגונים רבים אינם מעריכים נכונה את המאמץ הנדרש בתהליכי הגזירה והטיוב של הנתונים, ובסופו של דבר הם מוצאים את עצמם מופתעים. "חיתוך פינות" וויתור בשלב זה עלול לגרום לאיכות ירודה של נתונים במחסן הנתונים, ולאכזבה בקרב המשתמשים. נקודה נוספת שכדאי לציין היא שבדרך כלל לא ניתן לצפות מראש את כל בעיות איכות הנתונים. בדרך כלל, התהליך איטרטיבי, כך שלאחר תלונות מצד המשתמשים מאתרים בעיה חדשה, מתקנים אותה וחוזר חלילה. כך, עד לקבלת רמת איכות גבוהה של הנתונים הזורמים למחסן הנתונים.

גזירת הנתונים (Data Extraction)

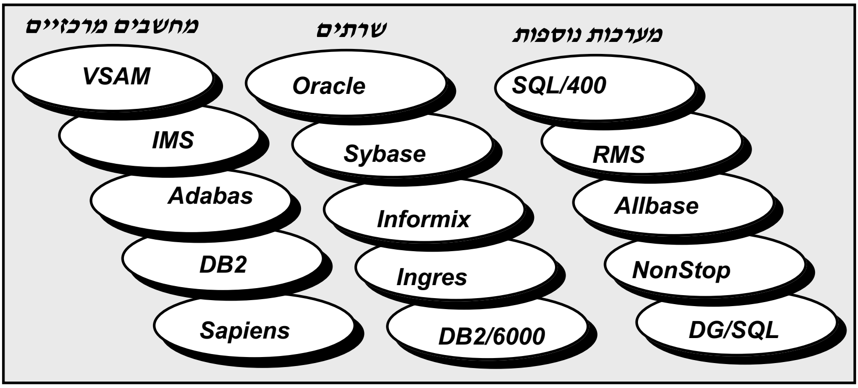
למעשה, הפעולה הראשונה שיש לבצע היא לגזור את הנתונים מהמקורות שלהם. מחסן הנתונים מוזן בנתונים המגיעים משני מקורות עיקריים:

- **מקורות פנימיים** (Internal Sources) - אלה נתונים שהארגון מנהל במסגרת מערכות המחשוב שלו. רוב הנתונים הרלוונטיים למחסן הנתונים מקורם ביישומים התפעוליים של הארגון. כפי שכבר הודגש קודם לכן, הנתונים התפעוליים יכולים לבוא ממספר יישומים שונים הפועלים באותו מחשב, או לעיתים במחשבים שונים, ובבסיסי נתונים שונים. גם אם הנתונים מנוהלים באותה פלטפורמת חומרה, עדיין יתכנו סתירות בין הנתונים, חוסר תאימות בפורמטים שלהם, שיטות ארגון שונות וכדומה.

- **מקורות חיצוניים** (External Sources) - אלה הנתונים שמקורם ממערכות ויישומים חיצוניים לארגון. לדוגמה, חברות המתמחות בבניית מאגרי מידע לגבי שווקים מסוימים, בנק ישראל, הלשכה המרכזית לסטטיסטיקה וכדומה. מקורות נתונים אלה חשובים מאוד למחסן הנתונים, כדי לאפשר למשתמש לקבל תמונה רחבה אותה לא ניתן לקבל מהנתונים הפנימיים.

למרות שפעולה זו נראית כפעולה פשוטה, היא יכולה להיות מורכבת למדי ודורשת מאמץ וכלים מיוחדים לביצועה. הנתונים במערכות התפעוליות מנוהלים במבנים המותאמים לצרכי מערכות אלו. מבנים אלה משקפים, לעיתים, פשרות ואילוצים שהתפתחו במשך שנים של תחזוקת המערכות.

אחד האתגרים המרכזיים בגזירת הנתונים נובע מהעובדה שהנתונים התפעוליים יכולים להיות מנוהלים בפלטפורמות מחשוב שונות ומגוונות (מחשבי יבמ מרכזיים, מחשבי AS/400, מחשבי Unix שונים, מחשבי Alpha וכדומה), ומנוהלים בקבצים פשוטים (כמו VSAM) או בבסיסי נתונים קנייניים (כמו CA-IDMS, IMS/DB, CA-Datacom, Adabas וכדומה) ועד לבסיסי נתונים טבלאיים (כמו Oracle, Informix, Sybase, DB/2 ואחרים). לכל פלטפורמה כזו יש כלי גישה וגזירה מיוחדים.



תרשים 9.3: דוגמאות למקורות נתונים.

הנתונים שמתקבלים ממקורות חיצוניים יכולים להגיע בפורמטים שונים. בדרך כלל, לארגון כמעט ואין שליטה בהם, ולכן יש לבנות תוכניות קריאה מיוחדות עבור מקורות אלה. הנתונים יכולים להתקבל כקבצים שטוחים במבנה ASCII, או בפורמטים מיוחדים כגון Lotus Notes, קבצי Excel וכדומה.

מדוע יש חשיבות לגזירת הנתונים מהמערכות התפעוליות

מי שלומד להכיר את תפיסת מחסן הנתונים נחרד להבין את רמת כפילות הנתונים הגבוהה שתפיסה זו יוצרת. לכאורה, מדובר כאן בהכפלת כל הנתונים שהארגון מנהל. מבט מעמיק יותר יכול להביא אותנו לידי מסקנה שלמעשה אין ברירה אחרת, למרות כפילות הנתונים הגבוהה. נסקור בקצרה את הסיבות העיקריות מדוע צריך תהליכי גזירה והמרה של הנתונים.

- **מבנה נתונים ייעודי** - מבנה הנתונים במערכות התפעוליות נבנה להשגת ביצועים טובים, בעת ביצוע מספר רב של תנועות עדכון. מבנה הנתונים במחסן הנתונים נוצר במטרה להקל על המשתמשים להבין את הנתונים, ולאפשר להם לנסח שאילתות ותהליכי חקירה באופן נוח, ללא מעורבות אנשי תוכנה. גזירת הנתונים יוצרת הזדמנות לביצוע המרה של מבנה הנתונים למבנה נתונים תואם לניתוחי מידע.

- **מעבר ממונחים תפעוליים למונחים עסקיים** - המונחים המקובלים במערכות התפעוליות פותחו בידי אנשי התוכנה, לשימושם בעת בניית היישומים. המונחים במחסן הנתונים חייבים להיות מונחים עסקיים ברורים ופשוטים, ולא מונחים טכניים.

- **ארגון המונחים** - מכיון שהמערכות התפעוליות מתפתחות לאורך זמן, עשוי להיווצר מצב בו קיימים מספר שמות שונים המתארים מונח אחד. במחסן הנתונים חייבים לבחור במונח המתאים ביותר, ושיהיה למונח אחיד עבור כל המשתמשים.

- ❖ **הפיכת מידע נסתר ועקיף למידע מפורש וגלוי** - אחת הטכניקות המקובלות במערכות תפעוליות היא להכניס משמעות לנתונים, לעיתים בצורה נסתרת. פעולה זו מתבצעת, למשל, על ידי שימוש בטווח מספרים להם משמעות עסקית. במחסן הנתונים צריך להשתדל ולהוציא משמעויות נסתרות אלו ולהפוך אותן לעמודות מפורשות וברורות.
- ❖ **סטנדרטיזציה בייצוג הפיסי** - בדומה לשימוש בשמות שונים, נמצא לעיתים שאותו מונח מיוצג באופן פיסי שונה במערכות התפעוליות. בעת העברת הנתונים אל מחסן הנתונים, חובה לבחור בייצוג אחד סטנדרטי.
- ❖ **סטנדרטיזציה בקודים** - מערכות תפעוליות שונות משתמשות בקודים שונים, בעלי אותה משמעות. בעת העברת הנתונים אל מחסן הנתונים, חייבים ליישר סתירות אלו, ולבחור בקודים אחידים ובעלי משמעות.

כעת נסקור מספר סוגיות, עליהן יש לתת את הדעת בעת עיצוב תהליכי הגזירה.

קריאת נתוני המקור

שלב זה עוסק בקריאת נתונים ממקורות שונים. למרות ששלב זה נראה פשוט, הוא עשוי להיות מורכב למדי, מפני שבארגון טיפוסי מספר מקורות הנתונים יכול להיות רב. לעיתים, יש לגשת לקבצים שטוחים, לבסיסי נתונים היררכיים או רשתיים או טבלאיים, או לשילוב כלשהו ביניהם. קריאת נתוני המקור מתבצעת תמיד במערכת המקור, בפלטפורמה בה היא פועלת, אך סביר להניח שצריך לקרוא נתונים מפלטפורמות חומרה שונות.

לכל אחד ממקורות הנתונים יש שיטות שונות לגישה לנתונים. אחד הדברים הבסיסיים ביותר בשלב זה הוא הכרת מבנה הנתונים, דבר הנראה מובן מאליו, אולם לעיתים מתברר שאינו כה פשוט. לא תמיד קיים תיעוד מסודר של הנתונים, לא תמיד ברור כיצד הם מתעדכנים, מה משמעות כל שדה בקובץ וכדומה. לעיתים, המצב אף גרוע יותר - כשהמערכת התפעולית מבוססת על חבילת תוכנה, שמבנה הנתונים שלה אינו ידוע (אלא רק לספק החבילה), והדרך היחידה לגשת לנתונים היא על ידי מחולל דוחות, או על ידי ממשקי התכנות (API) המסופקים על ידי יצרן חבילת התוכנה.

מתי לבצע את גזירת הנתונים

לכאורה, זו שאלה שהתשובה עליה פשוטה - בסוף יום העבודה. הבעיה היא שהמציאות מורכבת הרבה יותר. מצב בסיסי הנתונים התפעוליים בסוף יום עבודה, אינו בהכרח מצב תקין ועקבי. לעיתים, המערכות התפעוליות פועלות ללא הפסקה, והמושג **סוף יום עבודה** זר להן. דוגמאות למצב לא תקין ניתן למצוא במערכת האוספת נתונים מנקודות מכירה (POS). עד שלא נסיים את קבלת הנתונים מנקודת המכירה האחרונה, מצב בסיס הנתונים מוגדר כ**לא תקין**. האם נתחיל בתהליכי גזירת הנתונים, במקביל לתהליכי העברת הנתונים מנקודות המכירה, או שנמתין עד לסיום ההעברות?

סנכרון נתונים הנגזרים מקבצים שונים

לעיתים, לקבצים שונים יש קצב עדכון שונה, ונוצרת בעיית סנכרון לא פשוטה. לדוגמה, בחברת טלפונים, עדכון פרטי לקוח חדש בכל המערכות מתבצע רק בסוף יום, אולם הלקוח יכול להתחיל לבצע שיחות כבר במשך היום. אם נחליט לגזור את נתוני השיחות באופן שוטף, מייד עם הגעתן מהמתג הציבורי, נמצא עצמנו במצב בו מתבצעות שיחות ללקוח לא מזוהה.

שיטת הגזירה (Extraction Method)

בעיקרון, קיימות שתי שיטות גזירה שונות:

- **גזירה מלאה** - גוזרים את כל הנתונים מחדש בכל פעם (Full Data Propagation).
- **גזירה אינקרמנטלית** - גוזרים את השינויים בלבד (Changed Data Propagation).

ככל שפופולריות מחסן הנתונים גדלה, הופכת היכולת לשמור על עדכניותו לחשובה יותר. מחסני הנתונים הראשונים היו מבוססים בעיקר על גזירה מלאה של הנתונים התפעוליים, ובנייה מחדש של בסיס הנתונים בכל מחזור טעינה. תהליך זה מצטיין בפשטותו, אולם הוא יקר (מבחינת זמן הטעינה הנדרש), וככל שמחסן הנתונים מתפתח ומכיל יותר סוגי נתונים (וככל שנפח הנתונים בו הולך וגדל), הופכת גישת הגזירה המלאה ללא מעשית.

אחת השיטות להתגבר על הבעיה היא על ידי ניטור שוטף של המערכות התפעוליות, איתור השינויים בנתונים והעברת השינויים בלבד למחסן הנתונים. זו שיטת הגזירה האינקרמנטלית. היא מורכבת יותר, מפני שהיא דורשת יכולת לזיהוי השינויים שבוצעו בנתונים מאז הגזירה האחרונה.

המערכות התפעוליות הן דינמיות ומשתנות ללא הרף. השינויים יכולים להיות הן בתוכן בסיס הנתונים והן במבנה הנתונים עצמם. כאן נתמקד בשינויים בתוכן. בכל נקודת זמן מכילים בסיסי הנתונים התפעוליים רק את הגירסה האחרונה של הרשומות - כתובתו הנוכחית של הלקוח, יתרת מלאי נוכחית וכדומה (מפני שמנקודת ראות תפעולית אלה הנתונים החשובים ביותר). בחלק מהמקרים בהם נדרש עומק היסטורי, המערכות התפעוליות מנהלות יומן תנועות: תנועות מלאי, תנועות בחשבון, רשימת שיחות טלפון וכדומה. הצורך בזמני תגובה מהירים, ומגבלות בנפחי אחסון, גורמים לכך שכיום מקובל מאוד להגביל את העומק ההיסטורי של תנועות אלו - 10 תנועות אחרונות בחשבון, תנועות מלאי לחודש האחרון, שיחות טלפון לחצי השנה האחרונה בלבד, וכדומה. בדרך כלל עומק היסטורי זה מספיק לצרכים התפעוליים, אולם אינו מספיק כדי לעמוד בדרישות מחסן הנתונים.

יש להדגיש שבדרך כלל לא ניתן, ואף לא רצוי, להגדיר שיטת גזירה אחת בלבד. המציאות, בה קיימות מערכות התפעוליות הפועלות בפלטפורמות חומרה שונות, ועם מערכות קבצים וניהול בסיסי נתונים שונים, מאלצת אותנו לממש תהליכי גזירה שונים. תהליכים אלה מבוצעים בהתאם לסוג הקובץ, תדירות העדכון הנדרשת, מורכבות פיתוח שיטת הגזירה וכדומה.

כעת נסקור באופן מפורט יותר את שיטות הגזירה השונות הקיימות.

גזירה מלאה (Full Data Propagation)

בגישה זו, הנקראת לעיתים גם הגישה הסטטית, לוכדים תמונת מצב רגעית (Snapshot) של בסיס הנתונים, והיא אמורה להחליף את תוכן בסיס הנתונים במחסן הנתונים. מקובל להבחין בין שתי שיטות שונות:

- ❖ **טעינה מחדש (Reload)**: שיטה זו מניחה שניתן לבטל את תוכן הטבלאות השונות בבסיס הנתונים ולמלא אותן מחדש, על פי תמונת המצב העדכנית של בסיס הנתונים התפעולי. כדי לנסות ולהבהיר את השיטה, נשתמש בדוגמת טבלת הלקוחות. לאחר הטעינה הראשונית למחסן הנתונים, נוצר הצורך לעדכן את טבלת הלקוחות. הצורך נובע מכך שנוספו לקוחות חדשים, לקוחות שינו את פרטיהם, בוטלו לקוחות וכדומה. לפני ביצוע הגזירה מבטלים את טבלת הלקוחות במחסן הנתונים, ובונים אותה מחדש. הבנייה מחדש מתבצעת בעת תהליך הטעינה, מתוך תמונת המצב האחרונה של טבלת הלקוחות במערכת התפעולית.
- ❖ **הוספת שינויים (Append)**: בשיטה זו לא מבטלים את תוכן הטבלה הקיימת בבסיס הנתונים, אלא רק מעדכנים אותה, על פי תמונת המצב העדכנית של המערכת התפעולית. שורות חדשות מצטרפות לטבלה, שורות קיימות מתעדכנות על פי תוכן המערכת התפעולית, ושורות שאינן קיימות יותר במערכת התפעולית מבוטלות או מסומנות כלא פעילות.

זיהוי שינויים (Changed Data Identification)

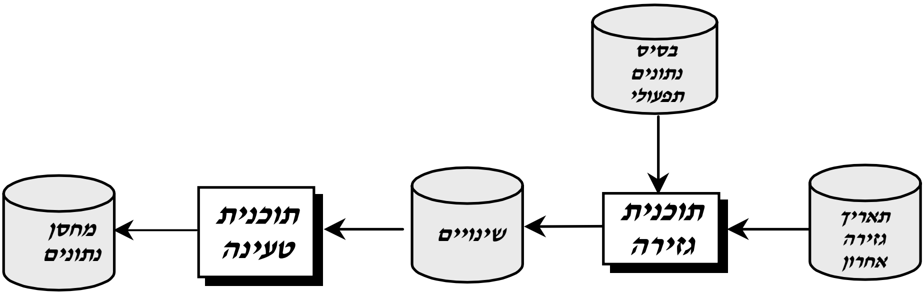
בגישה זו, הנקראת לעיתים גם **גישה אינקרמנטלית דינמית**, בונים ומשתמשים במנגנונים שונים. מטרתם של מנגנונים אלה לאתר ולזהות את השינויים שחלו בבסיסי הנתונים התפעוליים מאז הגזירה האחרונה, ורק שינויים אלה נטענים למחסן הנתונים. גישה זו מתאימה יותר לטיפול בטבלאות גדולות, מפני שאין צורך לטעון מחדש את כל הטבלה, אלא רק את השינויים. מצד שני, תהליך זיהוי השינויים הוא תהליך מורכב. לעיתים הוא דורש בניית מנגנונים ותוכניות יישום מיוחדות, ולעיתים יש לרכוש כלי תוכנה ייעודיים המסוגלים לזהות את השינויים.

נסקור את השיטות השונות לזיהוי השינויים בתוכן הנתונים במערכות התפעוליות:

- ❖ **חותמת זמן (Timestamp Capture)**: שיטה זו מניחה שכל רשומה בבסיס הנתונים התפעולי מכילה רישום של תאריך וזמן העדכון האחרון שלה. חותמת זמן זו משמשת כבסיס לתהליך הגזירה ומאפשרת איתור הרשומות שעודכנו מאז הגזירה האחרונה. בהנחה שתאריך הגזירה האחרונה נשמר בקובץ כלשהו, ניתן לבצע את הגזירה על ידי תוכנית יישום, הנכתבת במיוחד למטרה זו, או באמצעות תוכניות שירות של בסיס הנתונים.

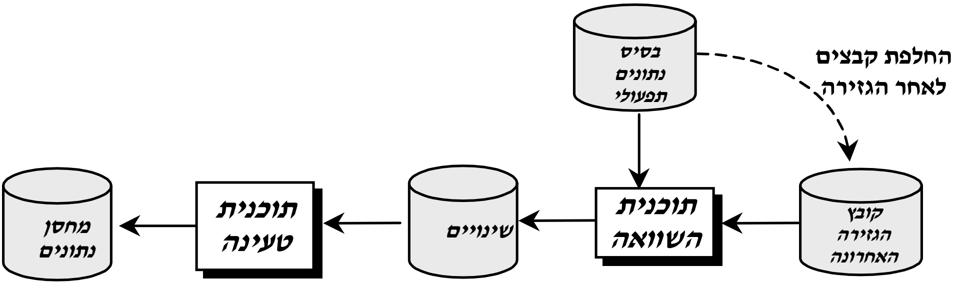
על פי שיטה זו אי אפשר לזהות מצב בו אותה רשומה עודכנה מספר פעמים מאז הגזירה האחרונה. לדוגמה, אם שורה מסוימת בטבלה עודכנה 5 פעמים מאז הגזירה האחרונה, תועבר למחסן הנתונים, על פי שיטה זו, רק תמונת המצב האחרונה. אם אין צורך בכל עדכוני הביניים, שיטה זו פשוטה ויכולה לפעול היטב. הבעיה העיקרית היא הצורך בקיום חותמת הזמן בכל השורות בטבלאות המערכת התפעולית. אם חותמת הזמן במערכות התפעוליות אינה קיימת, מחייב הדבר שינוי ביישומים ובמבנה הטבלאות, דבר מורכב מאוד עד בלתי אפשרי.

בעיה נוספת בשיטה זו הוא הטיפול בביטול שורות. אם המערכת התפעולית מבטלת שורה, תוכנית גזירת השינויים לא תמצא את השורה המבוטלת ולכן, באופן טבעי, גם לא תוכל לבטל אותה במחסן הנתונים. כדי להתגבר על בעיה זו אין לבטל שורות מהטבלאות התפעוליות, אלא רק לסמן אותן כבלתי פעילות. בשלב מאוחר יותר, בעת ארגון מחדש של בסיס הנתונים, ניתן לבטלן. אם המערכת התפעולית לא נבנתה מראש בצורה זו, יישום שיטת טיפול זו לביטול שורות כמעט ובלתי אפשרי.



תרשים 9.4: גזירת נתונים על בסיס חותמת זמן.

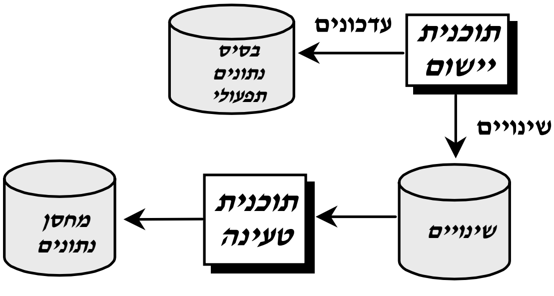
- **השוואת קבצים (File Comparison Capture)**: שיטה זו מניחה שהמערכת התפעולית אינה מכילה חותמת זמן, ולכן מבוססת על תהליך של השוואת שני קבצים - הקובץ האחרון שנגזר לעומת הקובץ הנוכחי. באמצעות השוואת שני קבצים אלה ניתן לזהות את כל השינויים מאז הגזירה האחרונה, כגון שורות חדשות שהצטרפו, שורות שבוטלו ושורות שבתוכנן חל שינוי כלשהו. שיטה זו מורכבת יותר משיטת **חותמת הזמן**, מפני שהיא מחייבת ניהול קובץ הגזירה האחרונה (שיכול לפעמים להיות קובץ גדול מאוד), ופיתוח תוכנית מיוחדת לאיתור השינויים. אחת האפשרויות הנוספות היא לא לשמור את כל קובץ הגזירה האחרונה. במקום זאת, צריך לשמור רק את מפתחות הרשומות שנגזרו. בדרך זו ניתן להקטין את נפח קובץ הגזירה שנשמר, לצורך ההשוואה הבאה. למרות מורכבותה היחסית של שיטת השוואת הקבצים, לעיתים זו השיטה המעשית היחידה.



תרשים 9.5: גזירה על ידי השוואת קבצים.

❖ **איתור שינויים נתמך יישום (Application Assisted Capture)**: שיטה זו מבוססת על ההנחה שניתן לבצע שינויים בתוכניות היישום התפעוליות, כך שתיצורנה קובץ נוסף, המכיל את כל השינויים המתבצעים בקבצים התפעוליים. מכיון שתוכניות היישום הן אלה המבצעות את השינויים, ניתן להתערב בלוגיקה שלהן במקומות המתאימים. בדרך זו גורמים לכך שבנוסף לביצוע העדכון בקובץ התפעולי, תירשם רשומה נוספת לקובץ השינויים.

לשיטה זו מספר חסרונות, מעבר לצורך בשינויים במערכות התפעוליות (דבר שלעיתים הינו מורכב מאוד, ולעיתים פשוט בלתי ניתן לביצוע). בנוסף, שיטה זו יכולה לגרום לירידה בביצועי המערכת התפעולית, בשל התקורה המתוספת כתוצאה מהצורך ברשומות חדשות אודות כל שינוי המתרחש. לפעמים, השינוי מתבצע במצב בו תוכנית היישום אינה מחזיקה בזיכרונה את כל פרטי הרשומה המתעדכנת, אלא רק את המפתח ומספר שדות רלוונטיים. אם רוצים לכתוב לקובץ השינויים את כל תוכן הרשומה שהתעדכנה, צריך לבצע גישה נוספת לקובץ, כדי להשלים את הפרטים החסרים. פעולה כגון זו עלולה לפגוע בביצועי המערכת עוד יותר.

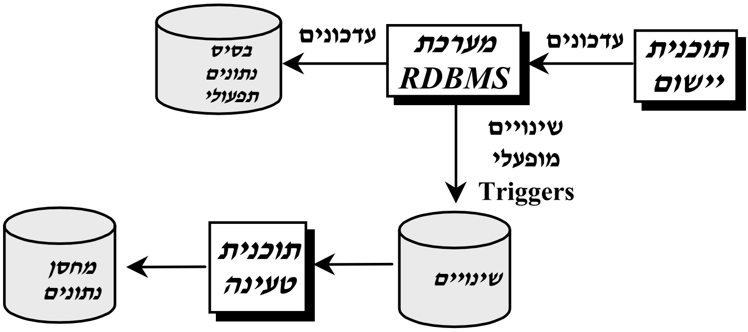


תרשים 9.6: בניית השינויים על ידי תוכניות היישום התפעוליות.

❖ **איתור מבוסס מזניקים (Triggers) של בסיס הנתונים:** מערכות RDBMS המודרניות מכילות מנגנון הזנקה (Trigger) המופעל באופן אוטומטי על ידי המערכת כתוצאה מאירועי העדכון השונים. למשל, ניתן להגדיר מזניק המופעל בכל פעולת הוספה, או בפעולת עדכון וכדומה. מנגנונים אלה, המופעלים אוטומטית במצבי עדכון, יכולים גם להכיל לוגיקה הכתובה בשפת תכנות ייעודית של יצרן מערכת

ה-RDBMS. ניתן להשתמש במנגנון זה כדי לרשום בטבלה חדשה את כל השינויים המתבצעים בטבלה התפעולית.

שיטה זו אינה דורשת התערבות בתוכניות היישום, מפני שהיא מתבצעת באופן חיצוני, באמצעות מנגנונים של מערכת RDBMS, ברמת הטבלה או ברמת העמודה. נשים לב ששיטה זו עלולה לגרום לירידה בביצועים, מאחר וכל שינוי בטבלה מפעיל את המנגנון, וגורם לכתיבת שורה נוספת בטבלת השינויים. קיימים גם מצבים בהם שינוי המתבצע בטבלה אינו מכיל את כל הפרטים הדרושים למחסן הנתונים. למשל, אם הטבלה היא טבלת יתרות המלאי, לא ניתן לבנות ממנה את תנועת המלאי שגרמה לעדכון היתרה.

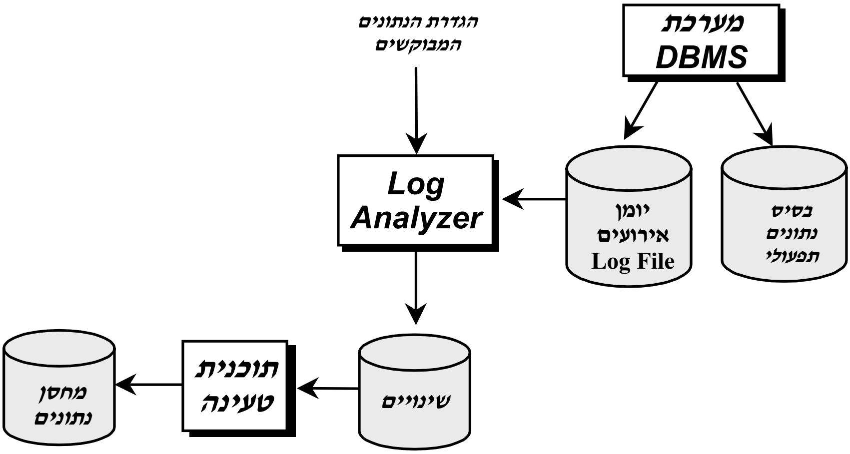


תרשים 9.7: שינויים מבוססי Triggers של בסיס הנתונים.

❖ **איתור מבוסס יומן אירועים (Transaction Log Capture):** שיטה זו מבוססת על העובדה שרוב מערכות RDBMS מכילות **קובץ יומן אירועים** (Journal או Log File). קובץ זה משמש לשמירה על שלמותה בסיס הנתונים, במקרה ומתגלה הצורך בגלגול חזרה של תנועות, או במקרה של תקלת יישום או נפילת חשמל וכדומה. מכיון שכל שינויי בסיס הנתונים ממילא נרשם בקובץ יומן האירועים, המנוהל על ידי מערכת ה-RDBMS, מדוע לא לנצל קובץ זה כדי לזהות את השינויים הרלוונטיים, ולגזור אותם לטובת מחסן הנתונים?

יצרני מערכת RDBMS ויצרני כלי תוכנה מצד שלישי פיתחו כלי תוכנה שונים הפועלים בסביבת בסיס הנתונים. כלים אלה מסוגלים לאתר את השינויים, באופן חיצוני ליישום, על ידי סריקה וניתוח שוטף של קובצי יומן אירועים (Log File Analyzer) וזיהוי הרשומות שעברו שינוי כלשהו. כלי תוכנה אלה יודעים לגזור רק תנועות תקינות, כלומר כאלה שביצעו פעולת Commit.

היתרון הבולט בשיטה זו הוא בכך שהיא חיצונית לתוכניות היישום ואינה מוסיפה תקורה לכל תנועה, אלא פועלת במקביל לבסיס הנתונים. ראוי לשים לב שתדירות סריקת יומן האירועים תהיה כזו שתבטיח את קיום כל השינויים, מכיון שבסיסי הנתונים מרעננים מעת לעת את יומן האירועים, כדי לחסוך בנפח האחסון שלו. כדאי לזכור שכלים אלה מותאמים לסוגים מסוימים של בסיסי הנתונים או קבצים, ואינם מתאימים לכולם. בקטגוריה זו ניתן למנות כלים שונים, כגון Replication Server של Sybase, או Data Propagator של יבמ.



תרשים 9.8: זיהוי שינויים באמצעות כלי תוכנה לניתוח יומן אירועים.

אז מהי שיטת הגזירה המומלצת

כפי שראינו בסקירה הקודמת, קיימות שיטות רבות לביצוע גזירת נתונים ממערכות תפעוליות למחסן הנתונים. הבעיה היא שאף אחת משיטות אלו אינה מושלמת, ואינה ניתנת להפעלה בכל מצב. לפני שמסיקים מהי שיטת הגזירה המומלצת, צריך לזהות באופן מדויק את מקור הנתונים: האם זהו קובץ או בסיס נתונים, האם ניתן לבצע שינויים בתוכניות היישום וכדומה. כפי שכבר הדגשנו, לכל מקור נתונים צריך לפתח שיטת גזירה מיוחדת. לכן, אין לדבר על שיטת גזירה אחת ויחידה, אלא על שיטת גזירה המותאמת למקור הנתונים.

הטבלה הבאה מציגה את שיטות הגזירה השונות, ומשווה בין מספר תכונות עיקריות בהן.

שיטת גזירה	השפעה על מקור הנתונים במערכת התפעולית	השפעה על תוכניות היישום	מורכבות היישום	יכולת לשמור על שינויים פרטניים	השפעה על ביצועי המערכות התפעוליות
גזירה מלאה	אין השפעה	אין השפעה	נמוכה	אין	נמוכה
חותמת זמן	השפעה נמוכה	השפעה נמוכה	נמוכה	אין	נמוכה
השוואת קבצים	אין השפעה	אין השפעה	גבוהה	אין	נמוכה
נתמך יישום	אין השפעה	השפעה גבוהה	גבוהה	קיימת	גבוהה
מבוסס Trigger	השפעה גבוהה	אין השפעה	בינונית	קיימת	גבוהה
מבוסס Log File	אין השפעה	אין השפעה	בינונית	קיימת	נמוכה

האם לשלב תהליכי מיטוב בתוך תהליכי הגזירה

אחת השאלות המורכבות בהן נתקל מעצב תהליכי הגזירה היא האם לנצל את תהליך הגזירה כדי לבצע מיטוב ובדיקות תקינות מסוימות של הנתונים, או לשחרר את תהליכי הגזירה מעומס מיותר. צריך לזכור שתהליכי הגזירה מתבצעים במחשבי הייצור של הארגון, ומחשבים אלה עמוסים במספר רב של מטלות. לכן, כל עומס נוסף שיוטל עליהם עלול לגרום לגלישה מחלון הגזירה העומד לרשותנו. מצד שני, קיימים מצבים בהם בדיקות מסוימות, ותהליכי מיטוב מסוימים, יכולים להתבצע רק מול המערכות התפעוליות. במקרים כגון אלה, הם חייבים להתבצע תוך כדי תהליכי הגזירה. בכל מקרה, העיקרון המנחה הוא להעמיס כמה שפחות את מחשבי הייצור, ולהשתדל להעביר את עומס בדיקות התקינות ומיטוב הנתונים אל סביבת מחסן הנתונים.

האם להעביר גם נתונים שלא נדרשים

מטרתנו להפוך את תהליך הגזירה לפשוט ככל הניתן, ולכן עולה השאלה האם לסלק מהרשומות המועברות שדות שאינם דרושים בבסיס הנתונים? ואולי כדאי להעביר את כל שדות הרשומה ולהשאיר את ההחלטה לגבי השדות הנדרשים בידי מחסן הנתונים? לא ניתן לקבוע כאן שיטה אחידה לכל המקרים. מצד אחד, אנו רואים את הפשטות של העברת כל שדות הרשומה, ומצד שני - את נפחי נתונים המועברים בין פלטפורמות מחשוב שונות ללא צורך. עם זאת, העברת כל נתוני הרשומה תאפשר בעתיד לטעון למחסן הנתונים גם נתונים שלא נדרשו בשלב זה, אבל ייתכן ויידרשו בעתיד. במקרה כגון זה לא צריך לשנות את תוכניות הגזירה.

כיצד לממש את תהליכי הגזירה

כפי שניתן לראות מסוגיות אלו, עיצוב תהליכי גזירה אינה משימה פשוטה. היא מחייבת מחשבה רבה המבוססת, בין היתר, על הבנה טובה של סביבת המערכות התפעוליות, מה האילוצים בה היא פועלת, מה מבנה הנתונים במערכות התפעוליות, מה קצב עדכון הנתונים, מה אמינות ואיכות הנתונים וכדומה. אחת הנקודות החשובות שיש לזכור בהקשר זה היא שלא מדובר במבצע חד פעמי. מכיון שהמערכות התפעוליות משתנות עם הזמן, יש לוודא שגם תוכניות הגזירה תעודכנה בהתאם. תחזוקה שוטפת זו של תוכניות הגזירה מוסיפה רובד של מורכבות, וחשוב שהארגון יהיה ערוך לכך מבחינת משאבים.

עם סיום עיצוב תהליכי הגזירה, עומדות בפנינו שתי חלופות למימוש:

- **כתיבה עצמית של תהליכי גזירה**: בחלופה זו לוקחת על עצמה יחידת המחשב את בניית התוכניות הנדרשות. במקרים מסוימים זו כמעט חלופה יחידה, מפני שלא קיימים כלים ייעודיים התומכים בכל הפלטפורמות, ובכל מבני הקבצים הקיימים. בחלק ניכר מהמקרים, גם אם קיימים כלים ייעודיים, עדיין מומלץ לבצע את הגזירה באופן עצמאי (לפחות בשלב הראשוני בפרויקט הקמת מחסן הנתונים), ולבחון את השימוש בכלים ייעודיים רק בשלב מאוחר יותר. בנוסף, צריך לזכור שעלות כלים אלה גבוהה יחסית, ולעיתים קשה להצדיק הוצאה זו בשלביו הראשוניים של פרויקט הקמת מחסן הנתונים.

❖ **שימוש בכלי גזירה ייעודיים**: בשל הקושי בפיתוח תהליכי הגזירה, והקושי בתחזוקה השוטפת של תוכניות אלו, פותחו מספר כלים ייעודיים שמטרתם להכניס מימד של מיכון (אוטומציה) בנושא זה. בין כלים אלה ניתן למנות את הכלים של חברת Prism, חברת Carleton ואחרות. כלים אלה מאפשרים להגדיר את כללי מיפוי הנתונים במחשב אישי המייבא אליו את הגדרות מבנה הנתונים, והם מחוללים באופן אוטומטי תוכניות בשפת תכנות כלשהי (Cobol, C). תוכניות אלו מותאמות למחשב היעד בו הן מיועדות לפעול.

בדיקת וניקוי הנתונים (Data Cleaning)

הנתונים המנוהלים במערכות התפעוליות יכולים להכיל "זיהומים" (תוצאה מבעיות שונות, כגון טעויות הזנה, העדר בדיקות תקינות, תקלות תוכנה וכדומה). הכנסת הנתונים מהמערכות התפעוליות ישירות למחסן הנתונים, ללא תהליכי ניקוי מתאימים, עלולה לגרום לבעיות קשות למשתמשים במחסן הנתונים. מטרת שלב זה היא להפעיל אלגוריתמים שונים לניקוי הנתונים, ולשמירה על איכותם.

נציג מספר דוגמאות לתהליכי ניקוי אפשריים.

בדיקת מרחב הערכים (Domain Inconsistency)

מטרת בדיקה זו לוודא שערכי הנתונים השונים נמצאים בתחום ערכים נכון. לעיתים, נמצא עמודות שהערכים המופיעים בהן אינם סבירים כלומר, הם מחוץ למרחב הערכים האפשרי של העמודה. לדוגמה, בטבלה הבאה השורה הראשונה מכילה תאריך לא תקין.

מספר לקוח	שם לקוח	כתובת	תאריך קנייה	קוד מוצר
30310	אייל הייפרמן	כצנלסון 1	10.13.97	AB320
30310	אייל הייפרמן	כצנלסון 1	25.5.97	CF-310-A
45210	שמעון אלקבץ	הרצל 25	22.5.97	AB320
46230	יואב הניג	החלוץ 10	2.10.97	XY-444-B
30310	אייל הייפרמן	כצנלסון 1	12.8.97	XY-444-B

הסיבות לטעויות בתחומי הערכים יכולות להיות רבות ומגוונות. מה שחשוב להבין הוא שהחלטות המבוססות על נתונים באיכות נמוכה עלולות להיות שגויות, ולכן עדיף להשקיע מאמץ באיתור תקלות אלו **לפני** טעינת הנתונים למחסן הנתונים. בדוגמה שבטבלה הקודמת, ייתכן שאיתור תאריך הקנייה האמיתי כבר בלתי אפשרי. לכן, יש לשקול כיצד לתקן אותו - תיקון החודש בתאריך, או קביעת ערך Null לשדה, כדי להצביע על כך שתאריך הקנייה למעשה אינו ידוע. צורת התיקון עשויה להיות שונה, תלוי בכל מקרה לגופו.

בדיקות מול טבלאות (Lookup Table Check)

לפעמים צריך לבצע בדיקת קיום ערך כלשהו מול טבלה מיוחדת, המכילה את אוסף כל הערכים החוקיים. למשל, בדוגמה הקודמת ייתכן שנחליט לבדוק את תקינות העמודה "קוד מוצר" מול טבלת קודי מוצר. ההחלטה לגבי ביצוע בדיקות מסוג זה היא לגופו של עניין, ובהתאם לרמת אמינות הנתונים המגיעים מהמערכות התפעוליות. אם מקור הנתונים הוא אמין ובדוק, בהחלט ייתכן שלא יימצא הצורך לבצע בדיקות אלו.

בדיקות אמינות נתונים (Data Integrity)

מטרת בדיקות אלו היא לבחון את אמינות הנתונים מבחינת המפתחות. הבדיקות המבוצעות הן בדיקת חד-ערכיות (Uniqueness) של המפתח העיקרי, ובדיקת אמינות הייחוס בין טבלאות (Referential Integrity). מומלץ לבצע בדיקות אלו בשלב זה, ולא בזמן טעינת הנתונים למחסן הנתונים. ביצוע בדיקות אלה בעת הטעינה עלול להאט מאוד את תהליך הטעינה ולגרום לאי עמידה בחלון הזמן העומד לרשותנו לצורך כך.

בדיקות לגילוי גרסאות שונות של נתונים (Version Check)

מבנה הנתונים במערכות התפעוליות משתנה כל הזמן, על פי צרכי המערכות התומכות בתהליכי העבודה השוטפים בארגון. לדוגמה, ייתכן שבמערכת התפעולית הוסיפו עמודות חדשות, שינו את אורך השדה, עברו מתצוגה נומרית לתצוגה אלפאנומרית, שינו את שיטת הקודים (למשל, קוד **מין** המכיל 1 או 2, הוסב לשיטת קידוד חדשה המכילה "ז" או "נ"). מכיון שלא כל שינוי במבנה הנתונים במערכות התפעוליות צריך להשתקף באותה צורה במחסן הנתונים, מטרת שלב זה לגלות את השינויים ולבצע את התיקונים (למשל, להחזיר את אורך השדה לאורכו הסטנדרטי שנקבע במחסן הנתונים, להעביר את הקוד החדש לקוד הישן וכדומה).

מיטוב נתונים (Data Enhancement)

שלב זה עוסק באוסף תהליכים, שמטרתם לשפר את איכות הנתונים לקראת הכנסתם למחסן הנתונים. שלב זה מורכב למדי, מכיון שהוא דורש בניית אלגוריתמים מורכבים. אלגוריתמים מורכבים אלו נועדו לאתר תקלות בנתונים, שמסיבות אלה ואחרות המערכות התפעוליות לא מנעו את התרחשותם. הדרך הנכונה לשיפור איכות הנתונים היא לדאוג לכך שהמערכות התפעוליות תבצענה את כל הבדיקות הדרושות. אולם דרישה זו לעיתים אינה מציאותית. הכנסת שינויים במערכות תפעוליות היא קשה, ולעיתים בלתי אפשרית.

בהמשך נציג רק מספר דוגמאות לתהליכי מיטוב נתונים. במציאות, כל ארגון צריך להגדיר מה מידת ההשקעה שמוכן הארגון להשקיע בתהליכי מיטוב אלה, וברמת התחכום שלהם.

הפיכת מידע נסתר ועקיף למידע מפורש וגלוי

אחת הטכניקות המקובלות במערכות תפעוליות היא להכניס לנתונים משמעות, לעיתים בצורה נסתרת. לדוגמה, מספר הלקוח, של לקוחות פרטיים מתחילים בקידומת 100, ואילו מספר הלקוח של לקוחות עסקיים מתחילים בקידומת 200. למעשה מספר הלקוח מכיל בחובו מידע לגבי סוג הלקוח (פרטי או עסקי), בנוסף לזיהוי הלקוח. בעת המעבר לסביבת מחסן הנתונים חובה להוציא משמעויות נסתרות ועקיפות אלו, ולהפכן לנתונים מפורשים. אחת התופעות הנפוצות היא שטווחים אלה מתמלאים, ולפתע מתברר שיש להקצות טווחים נוספים עבור אותה משמעות. לכן, לא יהיה זה מפתיע למצוא כעבור זמן מצב בו גם לקוחות שמספרם מתחיל בקידומת 201 הם לקוחות עסקיים. מפתחי התוכנה יודעים להתמודד עם מורכבויות אלו, אולם הם יכולים בקלות להפוך את השימוש במחסן הנתונים לסיוט. לכן, חובה לפשט מצב זה בעת גזירת הנתונים. בדוגמה הקודמת, עלינו ליצור בטבלת הלקוחות שבמחסן הנתונים עמודה נוספת, **סוג לקוח**, שתכיל את הקוד **פרטי** או **עסקי**. עמודה מפורשת זו מבטיחה לנו יציבות לאורך הזמן, וגם נוחות שימוש במחסן הנתונים.

איחוד רשומות (Record De-duplication)

כתוצאה מטעויות במערכות התפעוליות, ייתכנו מצבים בהם קיים חוסר תאימות בין נתונים השייכים לאותה יישות עסקית, כמו למשל **לקוח**. טעויות אלו עשויות לנבוע מכך שהמערכות התפעוליות אינן מבצעות בדיקות תקינות, חוסר תשומת לב מצד המשתמש המזין את הנתונים, טעויות בזמן הקלדת הנתונים, שינוי חלקי של נתוני לקוח ועוד. נתבונן בטבלה הזו:

מספר לקוח	שם לקוח	כתובת	תאריך קנייה	קוד מוצר
30310	אייל הייפרמן	כצנלסון 1	10.2.97	AB320
30310	אייל הייפרמן	כצנלסון 1	25.5.97	CF-310-A
45210	שמעון אלקבץ	הרצל 25	22.5.97	AB320
46230	יואב הניג	החלוץ 10	2.10.97	XY-444-B
47810	אייל הייפרמן	כצנלסון 1	12.8.97	XY-444-B

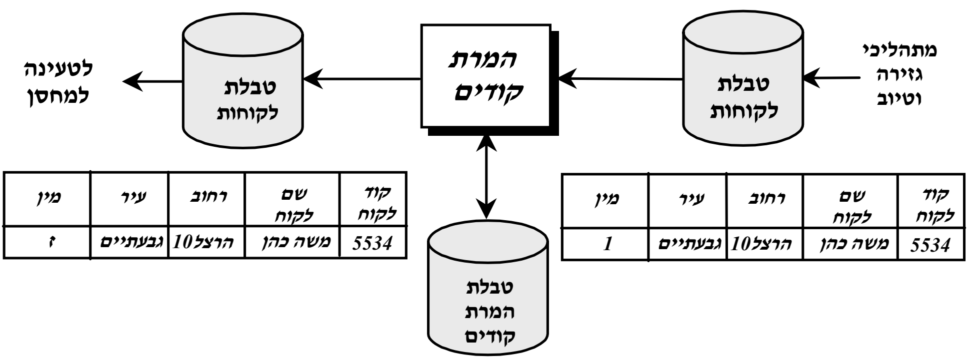
סביר להניח שהלקוח המופיע בשתי השורות הראשונות ובשורה האחרונה הם אותו לקוח. אולם, בשל טעות באיות השם, נפתחה רשומה חדשה עבור הלקוח, ולה מספר לקוח חדש. מסקנה זו נובעת מכך שיש דמיון פונטי בשם הלקוח וקיימת זהות בכתובתו. סביר להניח שזהו אותו לקוח, אולם בהחלט ייתכן גם שאלה שני לקוחות שונים. אם היינו יכולים להפעיל אלגוריתם ניקוי שמזהה מצב זה ונותן התראה, נוכל להשאיר את ההחלטה בידי מישהו שאחראי על תקינות הנתונים. אם ההחלטה היא שזהו אותו לקוח יש לבצע תיקון במערכת התפעולית, כך שלא נצטרך לבצע תיקון זה כל פעם מחדש בעת הכנסת הנתונים למחסן הנתונים. לאחר התיקון הטבלה תיראה כך:

מספר לקוח	שם לקוח	כתובת	תאריך קנייה	קוד מוצר
30310	אייל הייפרמן	כצנלסון 1	10.2.97	AB320
30310	אייל הייפרמן	כצנלסון 1	25.5.97	CF-310-A
45210	שמעון אלקבץ	הרצל 25	22.5.97	AB320
46230	יואב הניג	החלוץ 10	2.10.97	XY-444-B
30310	אייל הייפרמן	כצנלסון 1	12.8.97	XY-444-B

טעות תמימה זו יכולה להיות בעלת השלכות רבות על טיב ואיכות ההחלטות שיקבלו משתמשי מחסן הנתונים. לכן, חשוב לבצע תהליך ניקוי מסוג זה, למרות מורכבותו ועלותו הרבה (במונחי זמן עיבוד).

תקנון כתובות (Address Normalization)

לעיתים, מכילות המערכות התפעוליות כתובות שונות, כמו למשל כתובת לקוח או כתובת סניף, במבנה שם רחוב, מספר בית, עיר ומיקוד. כתובת זו אינה עוברת כל בדיקת תקינות לגבי מרכיביה (למשל, בדיקה שהעיר אכן קיימת, שהמיקוד חוקי וכדומה). אם קיימת דרישה לביצוע שאילתות שונות במחסן הנתונים על סמך נתוני הכתובת (למשל, רשימת הלקוחות באותה עיר, סך כל הגידול במכירות בעיר מסוימת לעומת עיר אחרת וכדומה), מסוכן להשתמש באופרטורים של מחרוזת (String Operators) כדי לחלץ את שם העיר. במקרים אלה, ניתן להפעיל אלגוריתמים מיוחדים המחלצים את שם העיר, מבצעים בדיקת תקינות של שמה מול טבלת ערים, ומכניסים את שם העיר לשדה נפרד ברשומה.



תרשים 9.9: תקנון כתובות כחלק ממיטוב הנתונים.

שינוי מבנה הנתונים (Data Transforming)

לאחר סיום תהליכי ניקוי ומיטוב הנתונים, ניתן להתחיל בביצוע תהליכי המרת הנתונים מהמבנה המקורי שלהם למבנה חדש. כזה המתאים למודל הנתונים המנוהל במחסן

הנתונים. ניתן להבחין בשני סוגי המרה שונים: המרה לקבלת מבנה נתונים פשוט ואחיד, ולמולה המרה לקבלת מבנה נתונים התואם את מודל הנתונים של מחסן הנתונים.

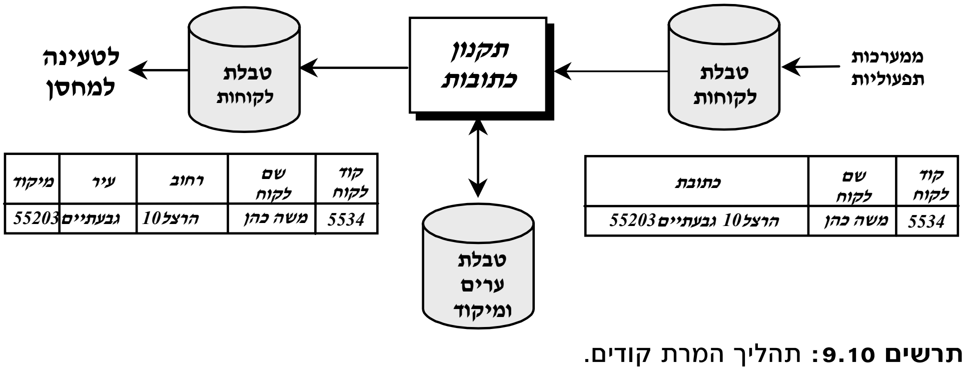
סטנדרטיזציה בייצוג הפיסי

לעיתים נמצא במערכות התפעוליות שאותו מונח מיוצג במספר צורות פיסיות שונות. לדוגמה, במערכת תפעולית אחת מספר הלקוח הוא שדה נומרי בן 12 תווים, ואילו במערכת תפעולית אחרת מיוצג אותו נתון על ידי שדה אלפאנומרי בן 13 תווים. בעת עיצוב מחסן הנתונים עלינו להחליט מהו הייצוג האחיד של המונח העסקי. למשל, נוכל להחליט שמספר לקוח יהיה 13 תווים אלפאנומריים. בעת שלב שינוי מבנה הנתונים נעביר את כל הייצוגים השונים של אותו מונח, לייצוג הסטנדרטי שנבחר.

שינוי קודים ואיחוד נתונים

שלב זה עוסק בהמרת נתונים לשם קבלת מבנה פשוט ואחיד. הדוגמאות הנפוצות להמרה זו הם: המרות קודים שונים לאוסף קודים אחיד, המרות הקשורות בקודי מטבע או יחידות מידה וכדומה. לדוגמה, אם מחסן הנתונים מוזן בנתונים המגיעים ממערכות תפעוליות הפועלות במדינות שונות, סביר להניח שכל מערכת כזו מספקת את הנתונים במטבע המקומי. כדי שניתן יהיה להפעיל שאילתות ופעולות סיכומיות, צריך לנהל את הנתונים במטבע אחיד כלשהו (למשל, ש״ח, דולר או אירו). במידה ונדרש ניתן לנהל את הנתונים גם במטבע המקור, בנוסף למטבע האחיד.

דוגמה נוספת להמרה בשלב זה יכולה להיות החלפת קודים שונים בקוד אחיד אחד. למשל, אם במערכת תפעולית אחת מופיע הערך **זכר** תחת קוד ״ז״, ובמערכת תפעולית אחרת מופיע אותו ערך תחת קוד ״1״ - יש לאחד את קוד הערך. בשלב ההעברה למחסן הנתונים צריך לקבוע את הקוד הסטנדרטי למין, ואז יש להחליף את הקודים השונים בקוד אחיד, למשל ״זכר״. גם אם בעיית קודים שונים אינה קיימת במערכות תפעוליות שונות, עדיין צריך לבצע המרות קודים. הקודים במערכות תפעוליות מנוהלים בקיצורים שונים, המתאימים למערכות תפעוליות, אך מאוד לא נוחים למשתמשי מחסן נתונים. לדוגמה, המשתמשים יתקשו מאוד לזכור שקוד ״**1**״ הוא עבור **זכר** והקוד ״**2**״ הוא עבור **נקבה**. כדי להקל על משתמשי מחסן הנתונים עדיף לבצע המרה של קיצורים אלה לקוד בו קל להשתמש ואשר אותו קל לזכור. למשל, ״ז״ עבור זכר ו-״נ״ עבור נקבה.



תרשים 9.10: תהליך המרת קודים.

העברת מונחים תפעוליים למונחים עסקיים

המונחים המקובלים במערכות התפעוליות פותחו על ידי אנשי התוכנה, לשימושם בעת בניית היישומים. לדוגמה, שדה המכיל את תעודת הזהות של הלקוח יכול להיקרא T-Z-Cust. למשתמש הקצה מונח זה חסר כל משמעות. גזירת הנתונים מהמערכת התפעולית לסביבת מחסן הנתונים יוצרת הזדמנות לתת לשדות שמות בעלי משמעות עסקית, למשל, Customer-Id בדוגמה שלנו.

ארגון המונחים

מכיון שהמערכות התפעוליות מתפתחות לאורך זמן נמצא, בדרך כלל, מספר שמות שונים לאותו מונח. למשל, במערכת אחת ייקרא מספר זהות של הלקוח T-Z-Cust, בשנייה Cust-Id ובשלישית Customer-Teudat_Zehut. תופעה זו נפוצה מאוד, מפני שהמערכות התפעוליות מתפתחות בזמנים שונים על ידי מפתחים שונים, ולא תמיד תוך משמעת ונוהל אחיד. גזירת הנתונים למחסן הנתונים יוצרת את ההזדמנות לקבוע את השם הסטנדרטי לכל מונח.

השלמת ערכים חסרים וברירות מחדל

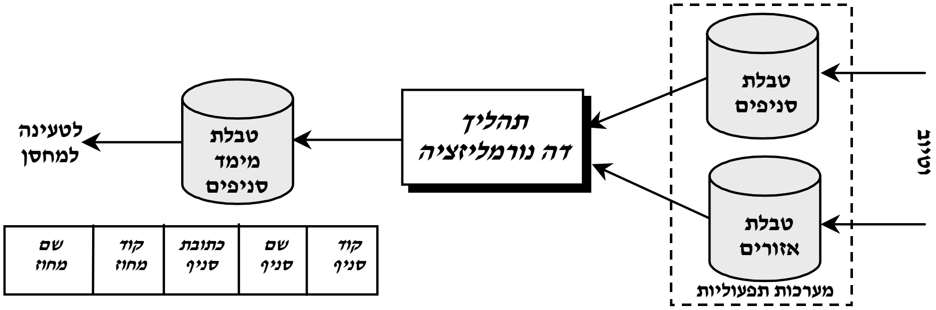
נתונים המגיעים מהמערכות התפעוליות אינם שלמים, ולעיתים חלק מהם חסר. דוגמה פשוטה ניתן לראות בשדה במערכת תפעולית, המציין מצב של **כן** או **לא**. למרות שמצב פשוט זה צריך להיות מתואר על ידי שדה שיכול להכיל שני ערכים בלבד, ניתן למצוא שדות כגון אלה המכילים שלושה ערכים - **כן**, **לא** ו**ריק**. הסיבה לשדה הריק היא שמפתחי התוכנה מבינים שערך חסר (ריק) משמעותו **לא**. אך מה בנוגע למשתמש הקצה? במצב זה, צריך להשלים את הערך הריק לאחד המצבים התקינים, ולא להעביר מורכבות זו למשתמש מחסן הנתונים. דוגמה נוספת ניתן למצוא בהזמנת רכש. נניח שיחידת המידה איננה שדה חובה ולכן צריך להשלים מידע חסר זה בעת העברת נתוני ההזמנות אל מחסן הנתונים. אם משתמש במחסן הנתונים יבקש לעשות ניתוחים שונים של ההזמנות על פי יחידות מידה, חשוב שנתון זה יושלם בשלב זה.

פיתוח הבדיקות והאלגוריתמים להשלמת נתונים חסרים, יכול להיות פשוט, אך לעיתים זו משימה מורכבת מאוד. לפעמים, קיים קושי בפיתוח אלגוריתמים אלה, ומעצבי מחסן הנתונים בוחרים באפשרות השלמת הנתונים החסרים על ידי ברירת מחדל סטנדרטית. בקביעת ברירות מחדל מסוג זה קיימת סכנה, מפני שהדבר עלול לגרום להסקת מסקנות מוטעות, בעת ניתוח הנתונים. לכן, עדיף להשתמש בסימון סטנדרטי כלשהו שיסמן שהנתון אינו ידוע (לדוגמה, להוסיף יחידת מידה מיוחדת **לא ידוע**).

דה-נורמליזציה של מבנה הנתונים (Data Denormalization)

מודל הנתונים המשמש את מחסן הנתונים שונה ממודל הנתונים של המערכות התפעוליות. בשלב זה מתבצעת ההמרה למבנה הטבלאות התואם את סכמת מחסן הנתונים, שבדרך כלל היא במבנה כוכב (Star Scheme) או במבנה פתיתי שלג (Snowflake Scheme). ההמרה לסכמות אלו מחייבת את בניית שני סוגי הרשומות העיקריים של מחסן הנתונים: רשומות עובדות ורשומות מימדים. בניית טבלת העובדות פשוטה יותר, ומבוססת על תהליך מיפוי פשוט בין הנתונים כפי שהם מגיעים ממערכות המקור אל מבנה רשומת העובדות. לעומת זאת, בניית רשומות המימדים היא פעולה מורכבת יותר ודורשת ביצוע תהליך דה-נורמליזציה.

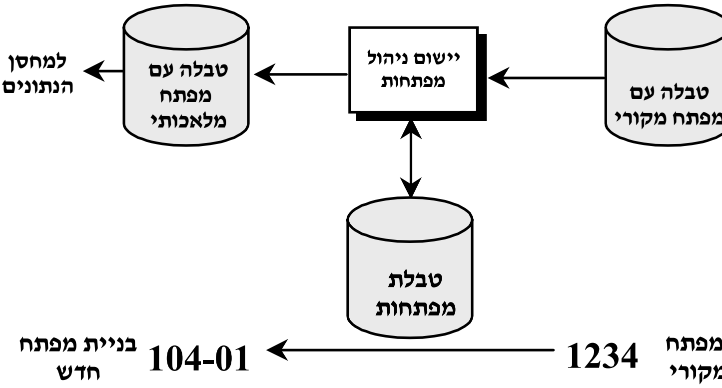
לדוגמה, בעת בניית טבלת הסניפים, צריך להכניס ברשומת הסניף את שם המחוז אליו שייך הסניף. טבלת האזורים אינה מקבלת ביטוי ישיר בסכמת מחסן הנתונים. לכן, היא מנוהלת רק בשטח הביניים בו מבצעים את המרות הנתונים. תוכנית יישום מיוחדת קוראת את רשומות קלט הסניפים שיש להעביר אל מחסן הנתונים, ועל סמך קוד המחוז מוסיפה את שם המחוז לרשומת הפלט.



תרשים 9.11: תהליך דה-נורמליזציה של נתונים.

בניית מפתחות מלאכותיים (Key Generation)

בשלב זה בונים את המפתחות המלאכותיים, עבור אותן רשומות בהן הוחלט שלא להשתמש במפתחות הטבעיים של הרשומות (כפי שהן מנוהלות במערכות התפעוליות). בדרך כלל, אלו רשומות המימדים אשר משתנות לאורך הזמן (Slowly Changing Dimensions). לדוגמה, אם הוחלט שלא להשתמש במפתח הסניף מהמערכת התפעולית, אלא לבנות מפתח מלאכותי, צריך לבנות יישום מיוחד שיקבע וינהל את המפתחות עבור טבלת הסניפים.

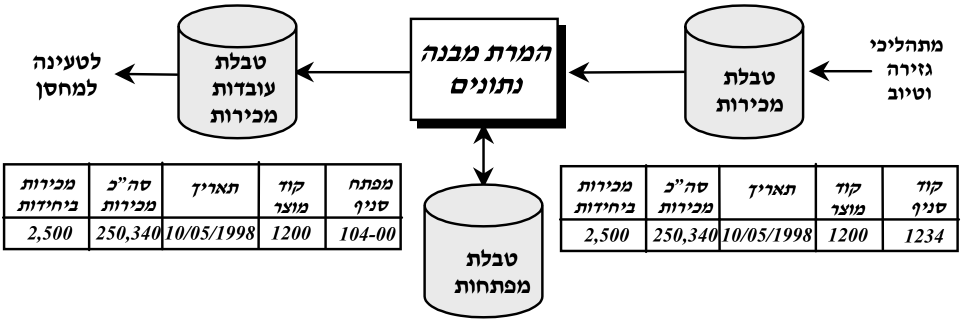


תרשים 9.12: בניית מפתחות מלאכותיים.

כפי שניתן לראות מהתרשים, יישום ניהול המפתחות מנהל טבלה מיוחדת, טבלת המפתחות בה נרשם המיפוי בין המפתח הטבעי לבין המפתח המלאכותי של הרשומה. בדוגמה המוצגת בתרשים, מפתח רשומת הסניף הוא "1234". יישום ניהול המפתחות (Key Administration Application) קבע שהמפתח המלאכותי של הרשומה יהיה "104-00". הסיומת 00 מציינת שזהו המפתח המלאכותי הראשון שרשומת הסניף קיבלה במחסן הנתונים. כשאחת התכונות של הסניף משתנה (למשל, השיוך שלו למחוז כלשהו), צריך לבנות רשומה חדשה לסניף, עם מפתח חדש ועם השיוך החדש שלו. במקרה זה ישתמש יישום ניהול המפתחות בטבלת המפתחות, ויקצה לסניף את המפתח "104-01".

המרת מפתחות עבור טבלת העובדות

בדרך כלל מגיעה טבלת העובדות מהמערכות התפעוליות במבנה קרוב למבנה המנוהל במחסן הנתונים. אם החלטנו לנהל מפתחות מלאכותיים עבור טבלת מימד אחת או יותר, צריך לבצע את המרת מפתחות המימד מהמפתח הטבעי, למפתח המלאכותי המתאים. לשם כך, יש לבנות תוכנית מיוחדת המשתמשת בטבלת המפתחות המנוהלת על ידי יישום ניהול המפתחות. טבלת מפתחות זו משמשת גם את היישום שמכין את רשומות טבלת העובדות. כל רשומת מכירה המועברת למחסן הנתונים עוברת המרת מפתחות מתאימה.



קוד סניף	קוד מוצר	תאריך	סה"כ מכירות	מכירות ביחידות
1234	1200	10/05/1998	250,340	2,500

מפתח סניף	קוד מוצר	תאריך	סה"כ מכירות	מכירות ביחידות
104-00	1200	10/05/1998	250,340	2,500

תרשים 9.13: הכנת רשומות טבלת המכירות.

בתרשים זה רשומת המכירה מתקבלת מהמערכת התפעולית, ונושאת את המפתח הטבעי של הסניף, המוצר ותאריך המכירה. מאחר ומפתח מלאכותי מנוהל רק עבור הסניף, מתבצעת המרה מהמפתח הטבעי 1234 למפתח המלאכותי 104-00. אם היינו מחליטים שגם עבור המוצר ינוהל מפתח מלאכותי, היה עלינו לבצע המרה גם עבור מפתח זה.

טעינת הנתונים למחסן הנתונים (Data Loading)

לאחר שקיבלנו נתונים נקיים ובמבנה תואם למבנה הנתונים במחסן הנתונים, ניתן להתקדם לשלב הבא ולבצע את טעינת הנתונים. המטרה - לבצע טעינה יעילה ככל הניתן ובפרק זמן הקצר ביותר האפשרי. מכיון שחלון הזמן העומד לרשותנו לתהליך הטעינה בדרך כלל קצר, יש לבצע כמה שיותר פעולות הכנה מקדימות ולפשט את תהליך הטעינה ככל הניתן. ניתן לחלק את תהליך טעינת הנתונים לשני שלבים:

❖ **טעינת הנתונים**: בשלב זה מתבצעת טעינת הנתונים אל טבלאות מחסן הנתונים. פורמט הנתונים המוכנים לטעינה הוא, בדרך כלל, ASCII ובמבנה התואם אחד לאחד לטבלאות בסיס הנתונים של מחסן הנתונים. ניתן להשתמש כאן בכלי הטעינה של יצרן בסיס הנתונים עצמו (למשל, SQL Loader של Oracle) העושה שימוש בעיבוד מקבילי לקיצור תהליכי הטעינה.

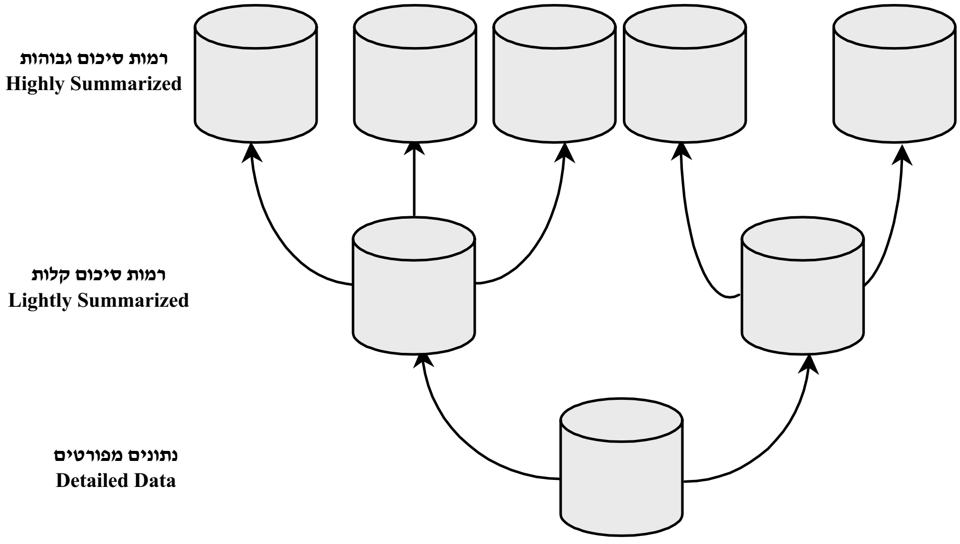
  שימוש בתוכניות עדכון רגילות, העושות שימוש בפקודות SQL, כגון Insert או Update, עלול לגרום להארכת משך שלב הטעינה. לכן, יש להשתדל לבצע את הטעינה באמצעות תוכניות השירות המיוחדות. מומלץ לבצע את טעינת הנתונים בעת הפעלת בדיקות הייחוס (Referential Integrity) בין הטבלאות. דבר זה יבטיח שהטבלאות החדשות אמינות, ושהקשרים בין הטבלאות (המבוססים על מפתחות זרים) תקינים. אמנם, תכונה זו מאיטה את קצב הטעינה, אולם היא מבטיחה את תקינות בסיס הנתונים. אם, בעת הטעינה מתגלות בעיות, יש לתקן רשומות אלו, במידת האפשר.

❖ **בניית האינדקסים**: לאחר שהנתונים נטענו למחסן הנתונים, מתבצע תהליך בניית האינדקסים לטבלאות. לעיתים, מקובל למחוק את האינדקסים הישנים ולבנותם מחדש, מפני שתהליך זה יעיל יותר מעדכון אינדקסים קיימים. גם כאן עושות מערכות RDBMS שימוש מלא בריבוי המעבדים וביכולות השרתים החדשים, כדי לייעל ולקצר את תהליך בניית האינדקסים.

תהליכי הזרימה מעלה (Up Flow Process)

חלק ניכר מניתוחי המידע המתבצעים במחסן הנתונים מתייחסים לנתונים מסוכמים ברמה זו או אחרת. למשל, מה היו סך כל המכירות בחודש זה לעומת החודש הקודם, מה מגמת השימוש בשירותי תא קולי ברבעון זה לעומת הרבעון המקביל בשנה קודמת וכדומה. אחת האפשרויות היא להכין סיכומים אלה בכל שאילתה, על פי הצורך. למשל, על ידי הגדרת טבלה מדומה (View) המכילה את הגדרת הסיכום. שיטה זו מאוד יקרה, מבחינת משאבי המחשב וביצועיו, ולכן מקובל להכין נתונים מסוכמים המנוהלים באופן קבוע במחסן הנתונים.

תהליכי הזרימה מעלה הם אוסף תהליכים העוסקים בבניית סיכומים (Summarization או Aggregation) בבסיס הנתונים. הטבלאות הסיכומיות הן אחד המנגנונים החשובים ביותר לקבלת ביצועים טובים, לשאילתות הפועלות מול מערכת RDBMS. ההנחה בבניית הסיכומים היא שרוב השאילתות פועלות רק בחלקים נבחרים של בסיס הנתונים. לדוגמה, שאילתה לגבי מכירות של מוצר מסוים, במהלך החודש האחרון, או כמות העובדים, לפי מקצועות, באגף מסוים בארגון. במקום לבצע סריקות בנפחי נתונים גדולים, יכול השימוש בטבלאות סיכומיות להקטין באופן משמעותי את משך זמן התגובה של השאילתה.



תרשים 9.14: תהליך הזרימה מעלה מנתונים שוטפים לנתונים סיכומיים.

כפי שניתן לראות מתרשים 9.14, ניתן לבצע סיכומים ברמות שונות. למשל, סיכום שבועי, חודשי, רבעוני או שנתי. כל רמת סיכום כזו יכולה להיות מנוהלת בטבלה מיוחדת, או שכל הסיכומים מנוהלים בטבלה אחת (עם אינדיקציה לרמת הסיכום בכל שורה). לכל שיטה כזו יש יתרונות וחסרונות.

הכנת טבלאות הסיכום

לפני שמתחילים בהכנת הטבלאות, יש לבטל את טבלאות הסיכום הקיימות. ביטול הטבלאות והאינדקסים שלהן יתבצע באמצעות פקודת SQL מיוחדות (Drop Table, Drop Index). הכנת הסיכומים החדשים יכולה להתבצע באופן אוטומטי, מייד בסיום תהליך הטעינה. לבניית הסיכומים ניתן לכתוב תוכניות יישום מיוחדות, או להפעיל פרוצדורות מוכנות, המכילות משפטי SQL. למשל:

```
1. CREATE TABLE TOTAL_SALES_0198 AS
2.     SELECT BRANCH_ID, PRODUCT_ID, SUM (SALE_PRICE * SALE_QTY)
3.     FROM    SALES
4.     WHERE SALE_DATE BETWEEN 010198 AND 310198
```

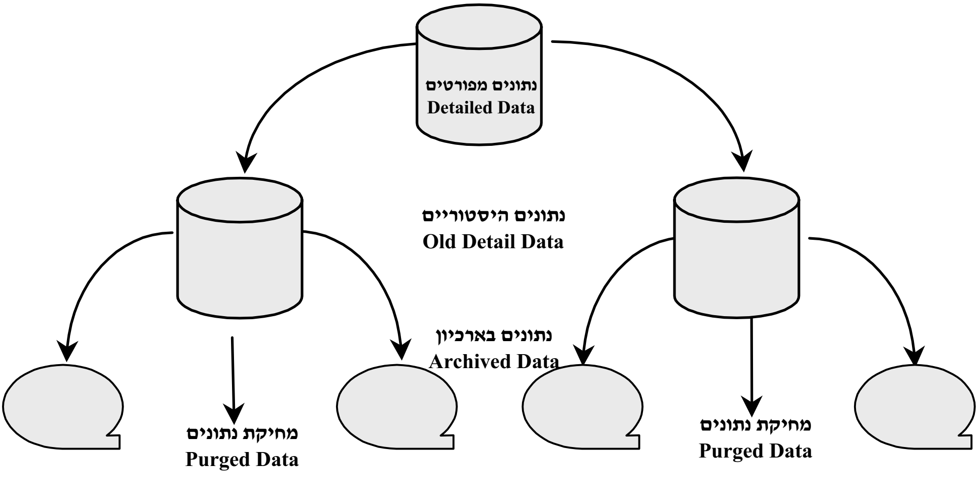
לעיתים, הכנת הסיכומים דורשת הפעלת לוגיקה עסקית, כגון סיכום כל החשבונות מסוג מסוים. הכנת סיכומים אלה מראש מסתירה מורכבות זו מעיני המשתמש, ולכן ניתן להתייחס למנגנון הסיכומים גם כאל מנגנון פישוט הנתונים וניתוחי המידע.

הכנת אינדקסים עבור טבלאות הסיכום

בגמר הכנת טבלאות הסיכום, יש לבנות את האינדקסים עבור הטבלאות. בניית האינדקסים מתבצעת באמצעות פקודות SQL, המיועדות לבניית אינדקסים - Create Index.

סקירת תהליכי הזרימה מטה (Down Flow Process)

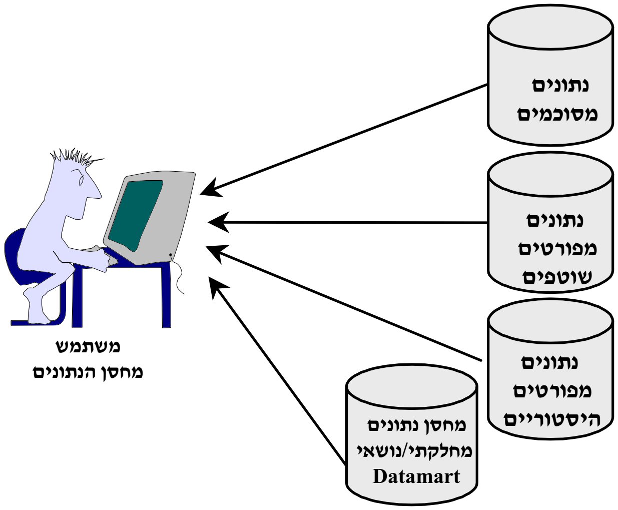
תהליכים אלה עוסקים בהעברה להיסטוריה וגיבוי (Backup Processes), ובמחיקת נתונים (Data Purging) ממחסן הנתונים.



תרשים 9.15: תהליך הזרימה מטה מנתונים שוטפים להיסטוריה.

סקירת תהליכי הזרימה החוצה (Out Flow Process)

תהליכים אלה עוסקים בכל הקשור לאספקת הנתונים ממחסן הנתונים אל משתמשי המחסן, ובעיקר בכל הקשור לבניית השאילתות (Query Processes). בנוסף לשאילתות עבור משתמשי מחסן הנתונים, עוסק תהליך זה גם בגזירת הנתונים ממחסן הנתונים אל מרכולי הנתונים, כלי כריית נתונים (Data Mining), כלי ניתוח רב-מימדיים (OLAP) וכדומה.



תרשים 9.16: תהליך הזרימה החוצה ממחסן הנתונים אל הצרכנים.

סקירת תהליכי הזרימה מידע על הנתונים (Meta Data Flow Process)

קטלוג נתונים פעיל (Repository), או **מילון נתונים** (Data Dictionary) כפי שהוא נקרא לעיתים, הוא רכיב חשוב ומרכזי במסגרת ארכיטקטורת מחסן הנתונים. קטלוג זה מכיל מידע אודות הנתונים (Meta Data) המנוהלים בבסיס הנתונים. המידע המנוהל במסגרת הקטלוג יכול להיות רב ומגוון (מה מקור הנתון, מה תהליכי הגזירה והטרנספורמציה שעבר הנתון, מה עדכניות הנתון, למי מותר לגשת אליו ועוד). בעוד שקיום קטלוג נתונים בסביבת מערכות תפעוליות יכול להיות בעל ערך ועזר, הרי שקיומו בסביבת מחסן נתונים הינו **כורח**. בלעדיו קשה מאוד למשתמש מחסן הנתונים להבין איזה נתונים מנוהלים, היכן, מה הקשר בין הנתון לנתונים אחרים ומה משמעותם. קטלוג הנתונים חייב להיות פעיל ולא סביל, כלומר עליו להתעדכן בזמן אמיתי בשינויים המתרחשים בבסיס הנתונים, או בשינויים בתהליכי גזירת הנתונים.

לקטלוג הנתונים שני תפקידים שונים, אך קשורים - תפקיד טכני ותפקיד עסקי.

- ❖ התפקיד הטכני בא לידי ביטוי בכך שהוא תומך בתפעול ובתחזוקת מחסן הנתונים. קטלוג הנתונים מנהל את המידע אודות מקור הנתונים, מה תהליכי הגזירה, המיטוב וההמרה שעוברים הנתונים, מהו מודל הנתונים של מחסן הנתונים, ממשקים לכלי CASE שונים, למתי הנתונים מעודכנים ועוד.
- ❖ התפקיד העסקי בא לידי ביטוי בתמיכה שמספק קטלוג הנתונים למשתמש הקצה, להבנת משמעות הנתונים וניתוחם. בהקשר זה פועל הקטלוג בדומה למילון נתונים מרכזי, המאפשר למשתמשי הקצה להתמצא במחסן הנתונים ולהבין את הפרשנות שיש לתת לנתונים.

כפי שנכתב קודם לכן, קטלוג הנתונים חייב להיות פעיל. כלומר, הוא מתעדכן בזמן אמיתי בכל השינויים המתרחשים. מצד אחד הוא חייב להיות משולב, באמצעות ממשקים שונים, עם כלי הגזירה והמרה, ואילו מצד שני הוא חייב להיות משולב עם כלי הקצה, המשמשים את המשתמשים. בפרק הבא נרחיב את הדיון על קטלוג הנתונים.

סיכום

פרק זה סקר את תהליכי הזרימה השונים המתרחשים במחסן הנתונים. כפי שניתן לראות מסקירה זו, קיימים תהליכים רבים, חלקם פשוטים יותר וחלקם מורכבים מאוד. עיצוב, בנייה וניהול שוטף של תהליכים אלה מהווה את אחד האתגרים הגדולים ביותר בבניית מחסן הנתונים. המאמץ הנדרש לממש תהליכים אלה מוערך ב-80% מהמאמץ הכולל לבניית מחסן הנתונים. לכן, יש חשיבות בהבנת האתגר ובבניית תהליכים המסוגלים להחזיק מעמד מעבר ליישום הראשוני של מחסן הנתונים.

פרק 10 : מידע על הנתונים (Metadata)

לשם מה דרוש מידע על הנתונים

כדי לאפשר למשתמשים לגשת למחסן הנתונים ולהפיק ממנו את התועלת המירבית, עליהם לדעת מה נמצא בו, מה משמעות הנתונים, איזה מערכות מקור סיפקו את הנתונים, נכון למתי הנתונים מעודכנים ועוד. מקובל לקרוא למידע זה **מידע על הנתונים**, או Metadata. מידע זה מספק את ההקשר, המאפשר את הפיכת הנתונים המנוהלים במחסן הנתונים לידע בעל ערך הדרוש בתהליכי קבלת ההחלטות. ברמה הבסיסית ביותר ניתן לומר ש**מידע על נתונים** הוא מידע המתאר את הנתונים ונותן להם את המשמעות.

כפי שהודגש בספר זה, מחסן הנתונים אינו מקום, אלא **ארכיטקטורה כוללת** המייצגת תפיסה המשלבת כלי תוכנה, חומרה, תהליכים ומשתמשים. כל כלי תוכנה מנהל אוסף מסוים של נתונים ומידע על הנתונים, כדי שיוכל לפעול ולבצע את עבודתו. דוגמאות למידע שיש לכלי התוכנה השונים יוכל להיות: מבנה רשומה של קובץ מסוים, קשרים לוגיים ופיסיים בין סוגי רשומות שונים, מאיזה בסיס נתונים נגזרים הנתונים, איזה מערכת הפעלה פועלת במערכת המקור ובמערכת היעד, כיצד רוצה המשתמש לראות נתונים מסוימים (Logical Views) ועוד.

בשל ריבוי הכלים המשמשים את סביבת מחסן הנתונים, ובשל האופי הטכני והלא אחיד במונחים, ברור כי יקשה על המשתמש לפעול בסביבה כזו. מטרת המידע על הנתונים היא לשמש כשכבת בידוד, המבודדת את המשתמש מהמונחים הטכניים של כלי התוכנה הפועלים בסביבת מחסן הנתונים. משתמשי מחסן הנתונים אינם מעוניינים במבנה ובפרטים הטכניים. הם רק רוצים לגשת במהירות ובנוחיות אל הנתונים, ומצפים לעשות זאת באמצעות המונחים העסקיים להם הם מורגלים. בנוסף להבנת המונחים ומשמעותם, רוצים המשתמשים גם לדעת את רמת עדכניות הנתונים, מהן המערכות מהן נגזרו הנתונים, מהם תהליכי ההמרה שעברו הנתונים וכדומה. רצוי שכל מידע זה ינוהל ב**קטלוג נתונים מרכזי** (Data Repository), כחלק ממחסן הנתונים וכחלק מהשירותים שהוא מספק למשתמשיו.

מהו "מידע על הנתונים"

הדרך הטובה ביותר להבהיר את המושג **מידע על הנתונים** תהיה באמצעות דוגמה. נשתמש באובייקט **מטוס** להדגמת סוגי מידע שונים שניתן לנהל לגביו. לכל אובייקט מופשט כזה אנו מצמידים אוסף תכונות המאפיינות ומתארות את האובייקט. לתכונות אלה מקובל לקרוא **מידע על האובייקט**. נדגיש, שהתכונות המתארות את האובייקט (בדוגמה שלנו ה**מטוס**) מתקיימות בנפרד מהאובייקט האמיתי.

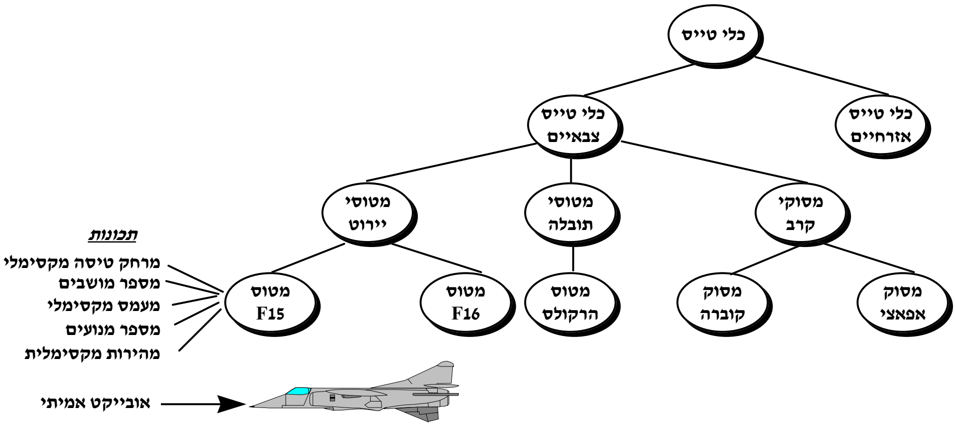
- **תכונות (Attributes) של אובייקט** - ברמה הבסיסית ביותר, לכל אובייקט יש אוסף תכונות המתארות אותו. לאובייקט **מטוס** יש תכונות כגון שם היצרן, דגם, שנת ייצור, מספר מנועים, מרחק טיסה מקסימלי, מספר מושבים בכל מחלקה, עומס מקסימלי ועוד. לאובייקט כמו מטוס יש מספר תכונות רב, ולכן ננהל בבסיס הנתונים רק את התכונות הרלוונטיות למרחב הבעיה בה אנו עוסקים.

- **סוג התכונה** - לכל תכונה מאוסף תכונות האובייקט נוכל לנהל מידע נוסף, כגון האם ערכי התכונה הם נומריים או אלפאנומריים, מה טווח הערכים וכדומה. לדוגמה, התכונה **מספר מנועים** יכולה לקבל ערכים נומריים בלבד שנעים בין 1 ל-5. התכונה **שם יצרן** יכולה לקבל ערכים מוגדרים מראש של יצרני מטוסים. יחידת המידה למרחק הטיסה המקסימלי, היא קילומטרים, וכן הלאה.

- **תכונות מולטימדיה** - לא כל תכונות אובייקט הן תכונות פשוטות, כמו אלה שהוצגו כאן. לדוגמה, למטוס מצורפים גם שרטוטי מערכות החשמל, שרטוטי המערכות ההידראוליות, שרטוטי מבנה המטוס, מסמכי אבטחת איכות שנוצרו בעת תהליך ייצורו ועוד. מידע נוסף זה יכול להיות מגוון מאוד - מסמכים, גרפים, מפות, שרטוטים, קול, וידאו וכדומה.

- **שיוך למחלקת אובייקטים (Object Class)** - עצם השימוש במילה "מטוס" כבר משייכת את האובייקט למחלקה מיוחדת של אובייקטים, על ידי הדגשת תכונה מרכזית, כלומר היותו כלי טייס.

- **היררכיות הפשטה (Abstraction Hierarchy)** - בדרך כלל אנו נוטים לבנות היררכיות הפשטה מורכבות, העוזרות לנו להבין ולשייך את האובייקט למחלקות ברמות גבוהות יותר. בדוגמה שלנו, נוכל לבנות רמת הפשטה מעל המחלקה "**מטוס קרב**". כעת, נשייך את מטוס הקרב לקבוצה "**מטוסי יירוט**", ואת קבוצת מטוסי היירוט לרמה הירכית נוספת, "**כלי טייס צבאיים**", המשלבת מטוסי יירוט, מטוסי תובלה ומסוקי קרב וכך הלאה. היררכיית הפשטה זו מספקת לנו ידע נוסף לגבי האובייקט, ולכן מקובל לראותה חלק מהידע הכולל אודותיו.



תרשים 10.1 : אובייקט אמיתי והירכיית הפשטה המתארת את האובייקט.

עד כאן דוגמת המטוס, שמטרתה היתה להדגיש את העובדה שלגבי כל אובייקט, או יישות, ניתן לנהל מידע רב ומגוון מאוד. חשוב לעשות את האבחנה בין המידע על האובייקט לבין האובייקט עצמו. בדומה לדוגמה הקודמת, לגבי כל יישות המנוהלת במחסן הנתונים (לקוח, הזמנה, חשבונית, פריט במלאי, תנועה בחשבון) אנו מנהלים מידע רב.

הבה נעבור כעת לדוגמה קרובה יותר לעולם מחסני הנתונים. נתבונן במחרוזת הבאה וננסה להבין את משמעותה.

12 233656 1234442

יהיה זה הוגן לומר שהמחרוזת חסרת משמעות מבחינתנו. היא יכולה להיות כל דבר, כמו למשל מספרי הזמנות לציוד, הכנסות שנתיות של אזרחים, או כל דבר אחר. הדרך היחידה לתת משמעות לנתונים היא על ידי צירוף ה**מידע על הנתונים** למחרוזת, כלומר הוספת שמות תכונות הנתונים. אם נוסיף את שמות התכונות, נראה שמשמעות המחרוזת הופכת לברורה.

מספר המחאה	מספר סניף	קוד בנק
1234442	233656	12

תרשים 10.2 : נתונים יחד עם המידע על הנתונים.

בעיקרון קיימות שתי דרכים לתת משמעות לנתונים :

- ❖ **צירוף** ה**מידע על הנתונים** אל הנתונים עצמם, כפי שהודגם קודם לכן.
- ❖ **גזירת משמעות הנתונים מהקשרם** (Context). למשל, אם היתה זו תוצאה של שאילתה בה ביקשנו פרטים מסוימים, משמעותם תהיה ברורה גם ללא צירוף **מידע על הנתונים**.

הדוגמה הקודמת היא המקרה הפשוט ביותר של **מידע על נתונים** המנוהל במחסן נתונים. מעבר לתכונות הבסיסיות של היישויות המנוהלות במחסן הנתונים, יש מידע רב מאוד אותו חשוב לנהל, כדי לאפשר למשתמשי מחסן הנתונים להפיק ממנו תועלת. במובן הרחב ביותר ניתן לומר ש**המידע על הנתונים** מתאר ומגדיר את כל סביבת הנתונים. לדוגמה, קובץ (או קבצים) בו מאוחסנים הנתונים, הקשר שלהם לתהליכים העסקיים מהם הם נובעים, מידע אודות תהליכי הגזירה וההמרה שעברו הנתונים וכדומה. ניתן להגדיר את **המידע על הנתונים** כאוסף כלל המידע אותו יודעים הכלים השונים הפועלים בסביבת מחסן הנתונים, אודות הנתונים וסביבת המחשוב. רמת השילוב בין הכלים השונים נקבעת, במידה רבה, ביכולת שלהם לשתף ביניהם את **המידע על הנתונים**. מידע זה יכול לענות על שאלות כגון:

- מה המשמעות העסקית של נתון מסוים.
- מה הערכים החוקיים שיכול לקבל השדה.
- איזה תהליכים גזרו את הנתון מהמערכות התפעוליות.
- מהם תהליכי ההמרה שעבר הנתון בדרכו למחסן הנתונים.
- מהו התהליך העסקי הנתמך על ידי אוסף זה של שאילתות.
- איזה תוכנית עדכנה את נתוני הלקוח בריצה האחרונה.
- איזו טבלה מכילה את נתוני המוצר, היכן היא נמצאת ומהו המבנה המפורט שלה.

כפי שניתן לראות מרשימה זו, המידע על הנתונים יכול להיות נרחב מאוד. את המידע על הנתונים נהוג לאחסן בקטלוג הנתונים (Repository), המהווה רכיב חשוב מסביבת מחסן הנתונים. בחלק מהמקרים, בשל העדר כלי ייעודי לניהול קטלוג הנתונים, מקובל להשתמש בכלי CASE כלשהו לאחסון המידע הרלוונטי. למרות שבמבט ראשון, נראה ניהול מידע זה טבעי ביותר, המציאות היא שבדרך כלל בניית קטלוג הנתונים היא אחת המשימות המורכבות ביותר בתהליך מימוש תפיסת מחסן הנתונים. אחת הסיבות לכך היא שהמידע אותו יש לנהל מגוון מאוד, קיימות סתירות בהגדרות הנתונים, חלק מההגדרות מסתתרות בכלי תוכנה, וכדומה. עצור לרגע וחשוב אודות המאמץ הנדרש להביא אנשים שונים להסכים להגדרת היישות "לקוח" - כל מחלקה בארגון מבינה יישות זו באופן שונה, ומשתמשת באוסף שונה של תכונות כדי לתאר אותה.

חשיבות המידע על הנתונים בסביבת מחסן הנתונים

בסביבות התפעוליות משמש **המידע על הנתונים** בעיקר את מנהלי בסיסי הנתונים, ואת מפתחי היישומים. הגישה לבסיסי הנתונים התפעוליים מתבצעת רק באמצעות יישומים מאפשרים למשתמשים השונים לבצע פעולות שונות בבסיס הנתונים. למשתמש הקצה (הפקיד בבנק, מנהל המחסן, איש המכירות, סוכן הנסיעות) אין כל ידיעה היכן, וכיצד הנתונים מאוחסנים. האינטראקציה שלהם עם הנתונים היא רק באמצעות מסכים ושאילתות, שהוכנו מראש על ידי מפתחי היישום.

סביבת מחסן הנתונים שונה באופן מהותי מהסביבה התפעולית. כאן המשתמשים ניגשים באופן ישיר אל בסיס הנתונים באמצעות כלי שאילתות, מחוללי דוחות, כלי ניתוח סטטיסטיים, כלי ניתוח רב-מימדיים וכדומה. הגישה שלהם היא בדרך כלל מזדמנת ומיועדת לפתור בעיות או לתמוך בתהליכי קבלת החלטות וכמעט ואין יכולת לבנות יישומים מוכנים מראש לנושאים אלה. כדי שמחסן הנתונים יהיה בעל ערך למשתמשים אלה, הם חייבים להבין את המבנה והתוכן שלו. מחסן נתונים ללא מידע על הנתונים דומה לארון מסמכים שבו המסמכים זרוקים ללא כל סדר והגיון. ברור שאיתור מסמך כלשהו במצב כזה כמעט בלתי אפשרית.

באנלוגיה ניתן לומר שמי שנוסע ברכבת אינו זקוק למפה, מכיון שהרכבת תוביל אותו אל היעד המבוקש. לעומת זאת, מי שנוהג במכונית בארץ זרה רצוי שיצטייד במפה, יבין את הסמלים השונים ואת תמרורי הדרך, אחרת לא יצליח להגיע ליעדו. לענייננו אנו, הנוסע ברכבת משול למשתמש ביישום מוכן מראש במערכת תפעולית, ואילו הנוהג ברכב בארץ זרה משול למשתמש במחסן הנתונים. מי שנוהג כל יום ממקום עבודתו לביתו אינו זקוק למפה, מכיון שהוא כבר מכיר את הדרך בעל פה. כך גם המשתמש המנוסה במחסן הנתונים. הוא ילמד להכיר את הנתונים ואת משמעותם, ולא יזדקק בכל פעם ל**מידע על הנתונים**. המשתמש המתחיל או שאינו מנוסה לא ימצא את דרכו ללא מידע זה. עם זאת, מכיון שדרישות המשתמשים לניתוח הנתונים משתנות כל הזמן וגם הנתונים במחסן הנתונים משתנים מעת לעת, גם משתמש מנוסה יזדקק ל**מידע על הנתונים**.

חוסר במידע על הנתונים, או מידע לא מדויק על נתונים, עלול להביא למצבים מוכרים בהם מנהל אחד בארגון טוען שהמכירות עלו ב-10%, בעוד שמנהל אחר טוען שהן ירדו ב-5%. מכיון שכל אחד מהם ניזון מהגדרות שונות ואולי גם ממקורות שונים, אין בכך כל פלא. עצם השימוש במחסן הנתונים כמקור אחיד של מידע לכל המשתמשים יכול לצמצם תופעות אלו. לעיתים, אחד האתגרים הראשוניים של פרויקט מחסן הנתונים הוא להגיע להסכמה על משמעות אחידה לנתונים ולמקורם (במידה וקיימים מספר מקורות אפשריים). קיום מידע על הנתונים במחסן הנתונים, מידע הקובע את הפרשנות הברורה והחד-משמעית של הנתונים הוא ערובה לכך שתופעות מסוג זה תעלמנה כמעט לחלוטין. המידע על הנתונים המנוהל במילון הנתונים חייב להיות אמין ומסונכרן עם הנתונים עצמם. אחריותו של צוות מחסן הנתונים היא לוודא אמינות שוטפת זו, ולמנוע מצבים בהם מילון הנתונים אינו משקף נאמנה את הנתונים המנוהלים בבסיס הנתונים.

מכל האמור לעיל ברור שהמידע על הנתונים בסביבת מחסן הנתונים הוא משאב בעל חשיבות רבה, לכל אורך מחזור החיים שלו. החל בשלב הבנייה, דרך שלבי הגישה של משתמשי מחסן הנתונים, וכלה בשלבי העדכון והתחזוקה השוטפים שלו.

קטגוריות של "מידע על הנתונים"

מהדוגמאות הללו ניתן לראות שהמידע על הנתונים יכול להכיל סוגים רבים של מידע. מקובל לחלק מידע על הנתונים לשלוש קטגוריות עיקריות:

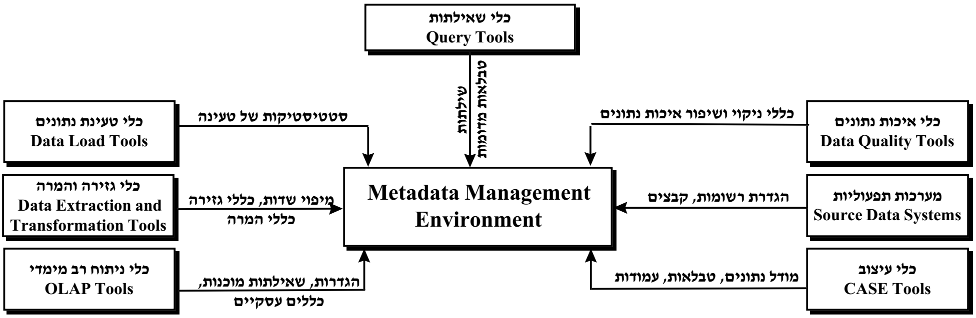
- **מידע על תהליכי ההמרה (Transformational Metadata):** מידע זה מגדיר מהם תהליכי ההמרה שעוברים נתוני המקור, וכיצד הם ממופים למחסן הנתונים. מידע זה מתייחס לכללי הגזירה, ההמרה, הניקוי, האינטגרציה וסיכום הנתונים. מקובל לצרף למידע זה גם מידע אודות עדכניות הנתונים (כלומר, למתי נכונים הנתונים), וכן מידע לגבי מקור הנתונים (כלומר, מאיזה קבצים ומערכות תפעוליות הם נגזרו). מידע זה משמש את צוות מחסן הנתונים בביצוע תהליכי המיפוי, הגזירה, ההמרה והטעינה. בשל כך קוראים לו לעיתים גם Back Room Metadata. בנוסף לצוות מחסן הנתונים, מידע זה הוא בעל חשיבות גם למשתמשי מחסן הנתונים. הוא מאפשר להם להבין מהיכן באו הנתונים, מהם תהליכי הגזירה וההמרה שהם עברו, למתי הם מעודכנים וכדומה.

- **מידע פיסי (Physical Metadata):** מידע זה מגדיר כיצד המאוחסנים נתונים במחסן הנתונים, ומכיל מידע הנובע מסכמת בסיס הנתונים. מבנה טבלאות בסיס הנתונים כולל את רשימת העמודות (ותכונות כל עמודה), ומידע אודות אילוצי אמינות בין הטבלאות (לדוגמה Referential Integrity בין הטבלאות, אילוצי ערכים של עמודות וכדומה). בדרך כלל מידע זה מעניין את אנשי המחשוב ומתוחזק על ידם, וכמעט ואינו מעניין את משתמשי מחסן הנתונים.

- **מידע עסקי (Informational and Business Metadata):** זהו המידע המעניין את משתמשי מחסן הנתונים ומתאר את מחסן הנתונים במונחים הבאים מעולמם. מידע זה משמש אותם בתהליכי השאילתות וניתוח הנתונים, ולכן מקובל לקרוא לו גם Front Room Metadata. המידע מתייחס למיגוון רחב של נושאים, ובא לענות על שאלות כגון: מה משמעותה של עמודה מסוימת, מאין מגיע נתון זה, כיצד מחושב הרווח הנקי, היכן ניתן למצוא את נתוני מכירות החודש האחרון, היכן השאילתה שבניתי בשבוע שעבר.

אם מחסן הנתונים כולל גם מידע חיצוני, יש חשיבות רבה בניהול המידע על נתונים אלה, כדי לאפשר למשתמשים לנצל מידע שאינו מוכר להם מפעילותם השוטפת עם הנתונים הפנים-ארגוניים. באופן כללי ניתן לומר שהמידע העסקי מסייע למשתמשים בעבודתם עם מחסן הנתונים. הוא מאפשר להם לגלות את משמעות הנתונים, וכיצד הם יכולים להשתמש בהם כדי לענות על שאלות עסקיות. אין כל ספק שמידע זה הוא בעל החשיבות הרבה ביותר בסביבת מחסן הנתונים. העדר מידע כזה, או קיומו החלקי, עלול לפגוע באופן משמעותי בהצלחת מחסן הנתונים בארגון.

מקורות המידע על הנתונים גם הם מגוונים, ויכולים לנבוע ממספר רב של כלים ומוצרים שונים.



תרשים 10.3: מקורות המידע על הנתונים.

נסקור בקצרה סוגים שונים של מידע על הנתונים, אותם מקובל לנהל במילון או גנזך הנתונים.

מיפוי וגזירת נתונים (Data Mapping and Extraction)

מידע אודות מיפוי הנתונים מציין את תהליכי המיפוי והגזירה שעברו הנתונים בדרכם מהמערכות התפעוליות אל מחסן הנתונים. מידע זה קושר, בדרך כלל, את מונחי מערכות המחשוב עם המונחים העסקיים ומתאר את המבנה הטכני של הנתונים. בדרך כלל, מידע זה הוא המפורט ביותר, והמדויק ביותר, ומשמש את משתמשי מחסן הנתונים, את מעצבי מחסן הנתונים ואת מנהלי הנתונים. בקטגוריה זו נמצא מידע כגון:

- ❖ מאיזו מערכת תפעולית נוצר נתון מסוים (למשל, ממערכת המלאי או ממערכת הכספית).
- ❖ זיהוי שדה המקור, ומיפוי בין שדה המקור ובין השדה במחסן הנתונים.
- ❖ תהליכי ההמרה שעבר הנתון (למשל, המרת הסכום ממטבע מקור למטבע אחיד, לפי שער חליפין ליום גזירת הנתון).
- ❖ שינויי מפתחות (ממפתח מקורי למפתח מיוחד במחסן הנתונים).
- ❖ ברירות מחדל של הנתון.
- ❖ הלוגיקה לבחירת הנתון, אם ייתכנו מספר מקורות (אם תאריך העדכון האחרון הוא בחודש האחרון, יש לדרוש את הנתון ממערכת המלאי, אחרת מהמערכת הפיננסית).

מיפוי למונחים עסקיים

מידע זה קושר, בדרך כלל, את מונחי המערכות התפעוליות עם מונחים עסקיים המקובלים על ידי המשתמשים.

- ❖ מיפוי שמות פנימיים של טבלאות או עמודות לשמות חיצוניים (Aliases): לדוגמה, עמודה בשם Qty_LY יכולה לשמש את מפתחי היישום לניהול נתוני המכירות בשנה אחרונה, אולם היא נראית חסרת משמעות, מנקודת מבטו של המשתמש במחסן הנתונים. ניתן למפות שם פנימי זה לשם חיצוני בעל משמעות, כגון Sales_Qty_Last_Year.
- ❖ משמעות הנתון מבחינה עסקית: מה זה מכירות נטו, האם המדובר ברווח תפעולי או נקי, האם הסכום **מכירות השנה** כולל מע״מ או לא, האם הרווח הוא בדולרים או בש״ח וכדומה.

מומלץ לקבוע שמות חיצוניים ברורים וקצרים, מפני שהם משמשים כפרמטרים בשאילתות וכותרות קבועות בתוצאות שאילתות ודוחות.

היסטוריית גזירת הנתונים

מחסן הנתונים מנהל מידע בעל עומק היסטורי. לכן, חשוב לנהל מידע הקשור להיסטוריית תהליכי הגזירה. בין היתר, יש לנהל מידע כגון:

- ❖ למתי הנתונים נכונים
- ❖ מה היו השינויים שהתרחשו במבנה הנתונים (למשל, אם לפני שנתיים בוצע שינוי בחלוקת סניפי המכירות לפי מחוזות, יש להימנע מביצוע השוואות של ביצועי הסניפים היום לעומת ביצועיהם לפני שנתיים)

טבלאות סיכום (Summarization Metadata)

בנוסף לטבלאות המכילות את העובדות הבסיסיות, כולל מחסן הנתונים גם מספר רב של טבלאות סיכום. סיכומים ברמות נמוכות (כגון, מכירות לכל שבוע) וסיכומים ברמות גבוהות מאוד (כגון, סך כל המכירות השנה). המידע המגדיר איזה סיכומים מנוהלים, כיצד הם מבוצעים, מתי בוצעו לאחרונה, מועדי עדכון וכדומה, הוא בעל חשיבות רבה למשתמש מחסן הנתונים. חלק מכלי השאילתות המודרניים עושים שימוש במידע זה בסיכומים המנוהלים במחסן הנתונים, כדי לנתב באופן אוטומטי את השאילתה לטבלאות המתאימות. כך ניתן לקבל את הביצועים הטובים ביותר.

מידע כמותי (Volume Oriented Metadata)

אוסף נתונים זה מספק למשתמש מחסן הנתונים מידע כמותי בחתכים שונים. למשל, כמה עמודות יש בטבלה, כמה שורות יש בטבלה, מהו אחוז השינוי בנפח הטבלה לאורך זמן, מה גודל הטבלה בבתים, כמה אינדקסים יש לטבלה וכדומה.

מידע אודות נגישות (Access Pattern Metadata)

המידע אודות מידת הנגישות לטבלאות השונות במחסן הנתונים, הוא בעל חשיבות. באמצעותו ניתן להחליט אילו טבלאות ניתן להעביר לאמצעי אחסון זולים יותר, ואילו טבלאות כדאי להעביר לאמצעי אחסון מהירים יותר.

בקרת גישות (Access Authorization)

מידע זה עוסק בהרשאות: למי מותר לראות איזה נתונים, באיזה תנאים, מאיזה מחשבים וכדומה. ניתן לכלול במידע זה גם מידע לגבי ניסיונות התחברות למחסן הנתונים, סטטיסטיקות שונות ברמת המשתמש וכדומה.

מידע מערכת (System Metadata)

מידע זה משמש בעיקר את בסיס הנתונים, ועוסק בנושאים הקשורים למבנה הפיסי שלו.

- גנזך בסיס הנתונים (טבלאות ה-System).
- מבנה המחיצות (Partitions).
- אינדקסים.
- הרשאות ברמת בסיס הנתונים.
- טבלאות מדומות (Views).
- פרוצדורות שמורות (Stored Procedures).

מוצרים לניהול המידע על הנתונים

המושג **מילון נתונים** או **קטלוג נתונים** (Repository) הוא מושג ותיק, ואינו קשור דווקא להופעת מחסני הנתונים. יצרנים שונים ניסו את כוחם בפיתוח מילוני נתונים כאלה ואחרים, כאשר היוזמה הידועה, ובמידה מסוימת היומרנית ביותר, היתה של חברת **יבמ** עם המוצר AD/Cycle. בפיתוח מוצר זה ניסתה חברת **יבמ** לבנות מילון נתונים רחב מאוד, אולם נטשה את המוצר מאחר ומכירותיו ויישומו בארגונים נכשל. תפיסת מחסני הנתונים הביאה לפריחה מחודשת של כלי ניהול מילון הנתונים, וזאת לאחר ניסיונות לא מוצלחים שונים להשתמש בכלי CASE, במחוללי יישומים דור רביעי לניהול מילון הנתונים.

בין החברות שפיתחו מוצרי ניהול מילון נתונים לסביבות מחסני נתונים, ניתן למנות את Prism, Carleton, ETI. כולן התמחו בנושא גזירת הנתונים מהמערכות התפעוליות. לכן, באופן טבעי, הכלים שלהם תומכים באופן מלא יותר בכל המידע לגבי מיפויים ותהליכי הגזירה. מוצרים אחרים, שבאו מעולם מילוני הנתונים, הם מוצרים של חברות אחרות, כגון Platinum, Rochade, IBM, IBI, Intellidex Systems, Logic Works ונוספות. נדגיש שכיום, כמעט ולא קיים מוצר אחד המסוגל לנהל את כל סוגי המידע על הנתונים אותם יש לנהל בסביבת מחסן הנתונים. לכן, שכיח למצוא יותר ממוצר אחד, כשכל מוצר מנהל סוג מסוים של מידע.

סיכום

אני מקווה שבנקודה זו כבר ברור לקורא מדוע כל כך קשה להגדיר את נושא המידע על הנתונים. במילים פשוטות ניתן להגדיר זאת כ**הכל חוץ מהנתונים עצמם**. פתאום נראה שניהול הנתונים הוא החלק הפשוט, בעוד נושא המידע על הנתונים הוא רחב מאוד, ולעיתים גם מורכב מאוד. כמעט כל כלי הפועל בסביבת מחסן הנתונים מנהל מידע על נתונים בצורה זו או אחרת. כפילות המידע המנוהל בכלים השונים יוצרת אתגר רציני לצוות מחסן הנתונים, שחייב לתאם ולסנכרן את המידע על הנתונים במספר רב של מוצרים. בשל מורכבות זו, חייב צוות מחסן הנתונים לשאול את עצמו: איזה מידע על נתונים לנהל, באיזה מוצר, כיצד יסונכרן מידע זה עם מידע המנוהל במוצרים אחרים ועוד. ללא ספק, אתגר מורכב שבשלב זה עדיין לא נמצאה לו תשובה פשוטה.

סיבוכיות זו הביאה לכך שמספר חברות מובילות בתחום כלי גזירה, כלי ניתוח רב-מימדיים, מערכות לניהול בסיסי נתונים ומערכות לניהול מילוני נתונים, הקימו קואליציה, The Metadata Coalition, במטרה לנסות ולהגדיר מספר תקנים שיאפשרו זרימה חלקה של המידע בין הכלים השונים. בשלב זה הם הגדירו מפרט עבור פורמט להחלפת מידע על נתונים (MIF - Metadata Interchange Format). ניתן לראות מפרט זה באינטרנט בכתובת **www.metadat.org**. בין החברות המובילות בקבוצה זו ניתן למנות את Microsoft, Business Objects, Prism, Arbor, ETI, Platinum, Cognos ואחרות.

רוב החברות מפתחות ממשק דו-כיווני באמצעותו ניתן יהיה לייצא או לייבא את המידע אל ומכלים אחרים. למפרט יש גם לא מעט מבקרים, הטוענים שהוא אינו מקיף דיו. זאת, למרות שמפתחי המפרט צמצמו אותו במכוון במהדורתו הראשונה, כדי שיישאר פשוט ובר יישום, ויענה בצורה סבירה על דרישות רוב סביבות מחסני הנתונים. חברת Oracle, אחת החברות המובילות בתחום מחסני הנתונים, החליטה להישאר מחוץ לקבוצה זו ולפתח תקן משלה. הנושא גדול ומורכב, ועדיין מוקדם להעריך מה מידת הצלחתה של קואליציה זו.

ללא תלות או קשר לפורמט אחיד זה, חלק מהחברות יצרו שיתופי פעולה ספציפיים כדי לאפשר שמידי יוכל לעבור בין הכלים שלהם. בשלב זה, שילוב המידע בין המוצרים השונים הוא במידה רבה אתגר לא פשוט, שצוות מחסן הנתונים צריך להתמודד איתו. בשל חשיבות המידע על הנתונים בסביבת מחסן הנתונים ניתן לומר שהזנחת נושא זה עלולה להביא לכישלונו של כל פרויקט מחסן הנתונים.

פרק 11: מרכולי נתונים (Data Marts)

מבוא

מערכות תומכות החלטה (Decision Support Systems - DSS) עברו מספר שלבים עד לגיבוש התפיסה המודרנית של מחסן הנתונים הארגוני. הניסיון שהחל להצטבר בבניית מחסני הנתונים גרם לארגונים להסיק מספר מסקנות, שהעיקריות בהן:

- ❖ הקמת מחסן נתונים ארגוני הוא אתגר מורכב ויקר
- ❖ הזמן הנדרש להקמתו ויישומו ארוך יחסית, בין היתר בשל משחקי הכוחות הפנים ארגוניים והבין מחלקתיים

קשיים ואתגרים אלה בהקמת מחסני הנתונים הארגוניים הובילו לתפיסה שונה, המבוססת על אותם עקרונות, אולם בעלת יעדים ושאיפות מוגבלות יותר: תפיסת **מרכול הנתונים - Data Mart**.

כפי שנפרט בפרק זה, הדרך הנכונה להתייחס למרכול הנתונים היא כאל **הרחבה טבעית** של תפיסת מחסן הנתונים לרמת הדרג המחלקתי. **אין** להתייחס למרכול הנתונים כאל תפיסה **חלופית** לתפיסת מחסן הנתונים הארגוני, אלא כאל תפיסה **משלימה**. בתחילה, היו ניסיונות להתייחס לתפיסת מרכול הנתונים כאל תפיסה המיועדת להחליף את תפיסת מחסן הנתונים, באמצעות פתרון מהיר וקל יותר. גישה זו התבררה כלא מוצלחת והניסיונות, בסופו של דבר, לא עמדו בציפיות ויצרו איי מידע חדשים המכילים נתונים סותרים ולא עקביים.

מטרת פרק זה למקד את הדיון בתפיסת **מרכול הנתונים**. לתפיסה זו יתרונות רבים, אולם קיימים בה גם סיכונים, כאשר הארגון אינו מבין את מלוא משמעותה. פרק זה יוצא מנקודת הנחה שהקורא מכיר את תפיסת מחסן הנתונים, כפי שהוסברה בפרקים קודמים, את שיטות העיצוב שלו ואת תהליכי העבודה הנהוגים בסביבת עבודה זו. כל אלה תקפים גם לגבי מרכול הנתונים, ולכן לא נחזור עליהם בפרק זה.

מהו מרכול נתונים

מרכול נתונים הוא מחסן נתונים ממוקד, המיועד לשרת מחלקה מסוימת או לתמוך בנושא יישומי מסוים.

קיימת הסכמה שמרכול נתונים הוא למעשה מימוש מסוים של תפיסת מחסן הנתונים, אבל בעל יעדים מוגבלים יותר. מעצם הגדרתו, ברור שמרכול הנתונים קטן יותר ממחסן הנתונים הארגוני ובדרך כלל הוא נגזרת חלקית כלשהי למחסן הארגוני. נגזרת זו ממוקדת בנושא מסוים, אשר בדרך כלל מעניין רק מחלקה אחת בארגון או מספר מצומצם של מחלקות קשורות. מרכול הנתונים ממוקד בנושא אחד, ולכן הוא צריך לתמוך במספר מצומצם יחסית של משתמשים. הוא צריך לנהל נפחי נתונים קטנים יותר (מספר עשרות עד מספר מאות ג'יגה-בתים) ומוזן בנתונים ממספר קטן יותר של מקורות.

מרכול הנתונים ממוקד וקטן יותר ולכן, תהליך הקמתו, עיצובו ויישומו מהיר יותר ועלות הקמתו נמוכה. מכאן גם הפופולריות הרבה שלו. חשוב להדגיש שעם הזמן, וככל שמרכול הנתונים צובר הצלחה ופופולריות בקרב משתמשיו, הוא יכול להתפתח ולתמוך בנפח הולך וגדל של נתונים ולשרת מספר גדל והולך של משתמשים.

להלן מספר דוגמאות למרכולי נתונים:

- ❖ מרכול נתונים שיווקי המתמקד בלקוחות ומכירות בלבד וישמש בעיקר את מחלקת השיווק והמכירות.
- ❖ מרכול נתונים בנושאי משאבי אנוש וממוקד בנתוני כוח אדם, תעסוקה, הדרכה וכדומה, וישמש בעיקר את מחלקת משאבי אנוש.
- ❖ מרכול נתונים המנהל את הנתונים הרפואיים של חולים שאושפזו במחלקה מסוימת של בית חולים, והמכילים נתונים ייחודיים לאותה מחלקה.
- ❖ מרכול נתונים בנושאי איכות השירות ללקוחות וממוקד בכל התקלות שנרשמו, איזה מוצרים אצל איזה לקוחות, ומשך הזמן "לסגירת" תקלות. מרכול זה ישמש את מחלקת אבטחת איכות, שיווק ופיתוח.
- ❖ מרכול נתונים של מחלקת ההנדסה בחברת טלקומוניקציה המיועד לניתוח התנועה ברשת התקשורת, לשם איתור מהיר של בעיות ברשת וכן למטרות תכנון קיבולת הרשת.
- ❖ מרכול נתונים בנושאי תמחיר, האוסף נתונים תמחיריים שונים ומשרת את מחלקת התמחיר בלבד.

כפי שניתן לראות מדוגמאות אלו, המשתמש הטיפוסי במרכול הנתונים הוא המשתמש המחלקתי שעוסק בתהליך קבלת החלטות ברמת המחלקה, ולא ברמת הארגון כולו.

הגורמים לפופולריות הרבה של תפיסת מרכול הנתונים

המציאות היא שמספר גדול של ארגונים החליט לממש את תפיסת מרכול הנתונים במחשבה (שלימים התבררה כמוטעית) שתפיסה זו באה להחליף את תפיסת מחסן הנתונים, ולפתור את רוב הבעיות והקשיים בהקמת מחסן הנתונים הארגוני. נסקור בקצרה את הסיבות העיקריות מדוע תפיסת מרכול הנתונים הפכה לנפוצה.

- **מבנה מותאם לדרישות מקומיות** - מאחר ומרכול הנתונים שייך למחלקה מסוימת וממוקד בנושא עסקי מסוים, יכולה המחלקה לבצע התאמות ושינויים מיוחדים, בעת הכנסת הנתונים למרכול הנתונים. מכיון שתיחום מרכול הנתונים הוא בגבולות המחלקה, ניתן להשתמש בו במונחים ייחודיים, ובהתאם לדרישות הייחודיות, ללא צורך בפתרון קונפליקטים בין מחלקתיים, הדורשים פתרון בעת הקמת מחסן הנתונים הארגוני.

- **עומק היסטורי קטן** - העומק ההיסטורי המנוהל במרכול תלוי רק בצרכי המחלקה. לכן, סביר להניח שהוא יהיה קטן יותר מדרישות ניהול העומק ההיסטורי במחסן נתונים ארגוני. מחסן נתונים ארגוני נדרש לספק ניתוחים ארגוניים מורכבים יותר.

- **תחרות קטנה יותר על משאבים** - בדרך כלל, מופעל מרכול הנתונים על החומרה של המחלקה. בשל כך, לא קיימת תחרות על משאבי המערכת מול מחלקות אחרות. במקרים מסוימים, מכיון שהבעלות על מרכול הנתונים ברורה, המחלקה יכולה לתקצב את הקמתו, ללא צורך בתחרות על משאבים כלל ארגוניים. כמו כן, קל יותר להצדיק את עלות הקמתו, ולהוכיח את התועלות שיספק.

- **שימוש בכלי ניתוח מותאמים יותר** - מכיון שהמחלקה היא בעלת המרכול, היא יכולה גם לקבוע באיזה כלי ניתוח מידע ברצונה להשתמש, בהתאם לדרישות משתמשיה. הכלים אינם חייבים להיות כלים שיענו על צרכים נרחבים, כמו הכלים הכלל ארגוניים.

- **בעיות אבטחת מידע מוגבלות יותר** - מכיון שהמידע במרכול הנתונים ממוקד ועומד לרשותה של מחלקה מסוימת, גם בעיית אבטחת המידע היא בעיה מצומצמת יותר. בעיה זו ניתנת לפתרון באמצעות כלים פשוטים יותר מאשר בעיית אבטחת המידע של מחסן נתונים ארגוני, המשרת מספר רב של מחלקות ומשתמשים ולכל אחד מהם הרשאה לראות רק חלק מהמידע.

- **עלות נמוכה יותר** - עלות הקמת מרכול נתונים קטנה באופן משמעותי מעלות הקמת מחסן נתונים ארגוני. החומרה מצומצמת יותר, היקפי הנתונים קטנים, מספר המשתמשים קטן יותר, מורכבות גזירת הנתונים קטנה יותר וכדומה. לכן, עלות הקמת מרכול הנתונים כולה נמוכה יותר. מקובל להניח שעלות ממוצעת להקמת מרכול נתונים עשויה לנוע בין מספר עשרות אלפים ועד מספר מאות אלפי דולרים, לעומת מספר מיליוני דולרים, עלות הקמתו של מחסן נתונים ארגוני.

❖ **הקמה מהירה** - מכיון שמרכול הנתונים ממוקד בנושא עסקי מסוים, קל יותר לעצב את מודל הנתונים, להגדיר את תהליכי גזירת הנתונים, מספר מקורות הנתונים מצומצם יותר. בסופו של דבר, משך הקמת מרכול נתונים קצר יותר ומסתכם בדרך כלל במספר חודשים, לעומת שנה ומעלה שדרושים להקמת מחסן נתונים ארגוני.

הסיכונים והבעיות בתפיסת מרכול הנתונים

הבעיה בתפיסת מרכול הנתונים היא שארגונים, לא מעט בעידודם של יצרני כלים שונים, החלו לראות במרכול הנתונים תפיסה שבאה להחליף את תפיסת מחסן הנתונים הארגוני. הם האמינו, בתמימות, שתפיסת מרכול הנתונים פותרת את רוב הבעיות, האתגרים והקשיים בהקמת מחסן הנתונים. מה שהתברר לרוב הארגונים הוא שתהליך ההקמה של מרכול הנתונים חייב להיות מבוסס על ארכיטקטורה רחבה יותר לאספקת מידע לקבלת החלטות בארגון. מי שלא נהג בדרך זו, מצא את עצמו עם אוסף של איי מידע מבודדים, המכילים נתונים כפולים וסותרים. אלה שלא התייחסו לארכיטקטורה כוללת, השקיעו משאבים רבים בבניית תהליכי גזירה והמרה כפולים של נתונים מהמערכות התפעוליות.

בסופו של דבר, ארגונים שנהגו ב"דרך לא דרך" החטיאו את אחת המטרות העיקריות שבתפיסת מחסן הנתונים: בניית סביבת מחשוב תומכת החלטות, אשר יכולה להציג תמונת מידע אינטגרטיבית ואמינה של הארגון. אלה סכנות שארגון חייב להבין היטב, לפני שהוא מקבל את ההחלטה לאפשר לכל מחלקה להקים מרכול נתונים משל עצמה.

כעת נדגיש את הבעיות והסיכונים העיקריים הקשורים במימוש תפיסת מרכול הנתונים כתפיסה חלופית לתפיסת מחסן הנתונים הארגוני.

❖ **חוסר עקביות בנתונים** - מכיון שסביר להניח שבמשך הזמן יפותחו ויתופעלו מספר מרכולי נתונים, סביר גם להניח שחלק מהנתונים ינוהלו במספר בסיסי נתונים שונים. לדוגמה, נתוני הלקוח במרכול השיווקי וכן במרכול העוסק בדיווחי תקלות מהשדה ומיועד לטפל בניתוחי איכות השירות. כמו בכל מצב בו יש כפילות נתונים, קיים סיכוי לחוסר עקביות בין הנתונים ולפעמים אף לסתירות. יש להשקיע תשומת לב רבה ומאמצים להקטין ככל שניתן סתירות אלו. במובן מסוים, מסוכן לחזור למצב בו לכל מחלקה יש תמונת מידע שונה ובישיבות בין מחלקתיות מתחילים בוויכוחים על נכונות הנתונים, ולא על אופן פתרון הבעיה.

❖ **כפילות בתהליכי הגזירה** - בשל הצורך לגזור נתונים עבור מספר מרכולי נתונים, נוצר מצב שאותו קובץ תפעולי עובר מספר רב של תהליכי גזירה. מעבר לחוסר היעילות ולבזבוז משאבי המחשב שגורמים תהליכי גזירה כפולים אלה, יש בכך גם סכנה להכנסת סתירות בשל תנאי גזירה שונים וזמני גזירה שונים.

❖ **מורכבות בהפצת הנתונים** - שכבת אחסון ביניים והפצה מקבלת בסביבה זו משנה חשיבות, בשל הצורך להפיץ את הנתונים למספר מרכולי נתונים שונים שיכולים להימצא גם באתרים גיאוגרפיים שונים. תהליכי ההפצה עצמם יכולים להיות מורכבים, ולעיתים מבוססים על כלים ייעודיים להפצת נתונים להרבה אתרים ובסיסי נתונים שונים.

❖ **טכנולוגיות שונות** - מכיון שמרכולי הנתונים מפותחים בזמנים שונים, ולפעמים על ידי צוותים שונים, נוצרת הסכנה שכל מרכול ישתמש בטכנולוגיה שונה, יגדיר את משמעות הנתונים באופן שונה, ובסופו של דבר יתקבל "מגדל בבל". מסיבה זו יש חשיבות לכך שארגון הבוחר לממש ארכיטקטורת מרכולי נתונים יגבש את התפיסה הכוללת, ויפקח על הסטנדרטים והטכנולוגיות. אסור לארגון לתת יד ליצירת אנדרלמוסיה במערכות תומכות ההחלטה שלו.

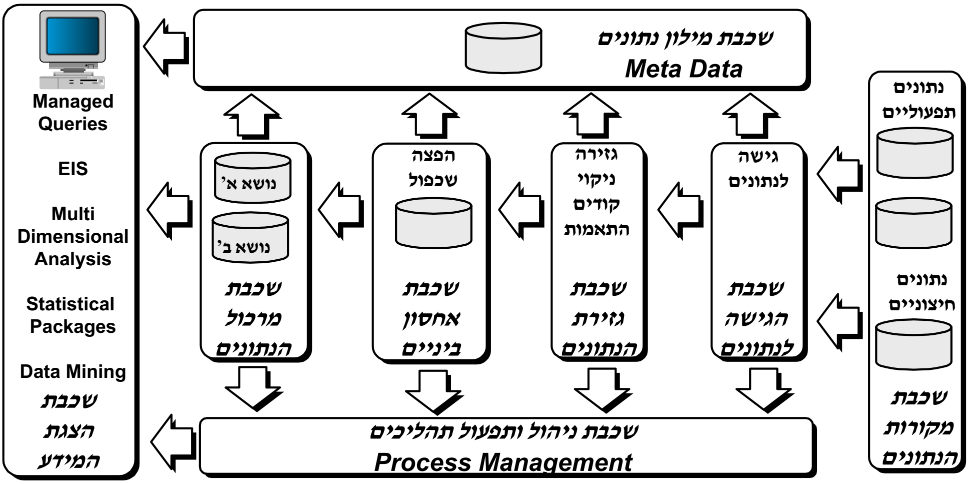
מכל האמור לעיל, נובע שהדרך הנכונה להתייחס למרכול הנתונים היא כאל **הרחבה** של תפיסת מחסן הנתונים לדרג המחלקתי. זו אינה תפיסה שבאה **להחליף** את מחסן הנתונים הארגוני. הבנה זו יכולה לחסוך הרבה כסף, משאבים ומאמצים.

ארכיטקטורת מרכול הנתונים

בעיקרון, ארכיטקטורת מרכול הנתונים דומה לארכיטקטורת מחסן הנתונים הארגוני, בהבדל מהותי אחד - בעוד שמחסן נתונים ארגוני יש רק אחד, מרכולי נתונים יכולים להיות רבים. אחת ההחלטות הבסיסיות ביותר שעל הארגון לקבל היא האם ברצונו להתבסס על ארכיטקטורה דו-שכבתית, בה מרכולי הנתונים מוזנים ישירות מהמערכות התפעוליות, או האם ברצונו להתבסס על ארכיטקטורה תלת-שכבתית, בה מרכול הנתונים מוזן ממחסן הנתונים הארגוני.

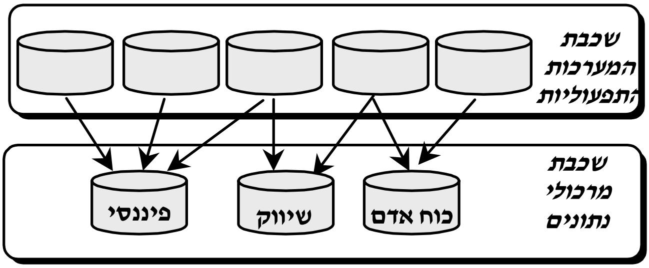
מרכול נתונים בארכיטקטורה דו-שכבתית (Two Tier Data Mart Architecture)

התרשים הבא מציג את הארכיטקטורה הכללית של מרכול נתונים, המבוסס על ארכיטקטורה דו-שכבתית.



תרשים 11.1: מרכול נתונים בעל ארכיטקטורה דו-שכבתית.

כפי שניתן לראות מהתרשים, כל השכבות שקיימות במחסן הארגוני, קיימות גם כאן. מכיון שגם בארכיטקטורה זו קיימת אבחנה בין שכבת הנתונים התפעוליים לבין שכבת הנתונים המיועדים לתמיכה בקבלת החלטות, ארכיטקטורת מרכול הנתונים היא ארכיטקטורה דו-שכבתית, מבחינת הנתונים.



תרשים 11.2: שכבות בארכיטקטורה של מרכולי הנתונים.

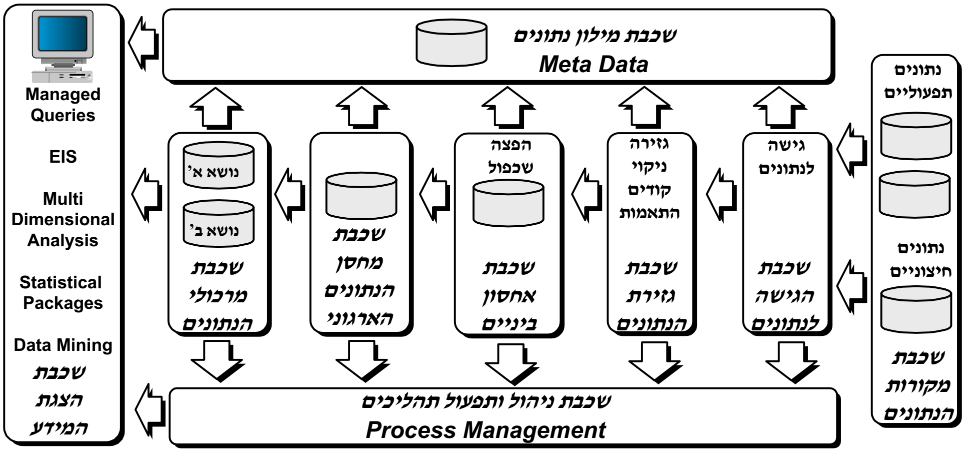
הארגון יכול להחליט על הקמת מרכול נתונים בארכיטקטורה דו-שכבתית בשני אופנים שונים:

- **מרכול נתונים מבודד (Isolated Data Mart)** - זהו מרכול נתונים המבוסס על מודל נתונים המשמש את המחלקה בלבד. מודל נתונים זה נבנה מתוך ראייה מקומית בלבד.
- **מרכול נתונים אינטגרטיבי (Integrated Data Mart)** - זהו מרכול נתונים המבוסס על מודל נתונים ארגוני. כלומר, מודל הנתונים נבנה מתוך ראייה גלובלית של כלל הנתונים שינוהלו במספר מרכולי נתונים מחלקתיים. הנתונים נגזרים מהמערכות התפעוליות ישירות לכל אחד ממרכולי הנתונים המחלקתיים.

נקדים ונאמר שתפיסת מרכול הנתונים המבודד מסוכנת מאוד, ולכן אינה מומלצת. למרות שארכיטקטורת מרכול הנתונים האינטגרטיבי נראית, על פניה, כארכיטקטורה מתאימה, חשוב להבין שהיא קשה מאוד למימוש. הפיתוי להשתחרר מהארכיטקטורה הארגונית גבוה.

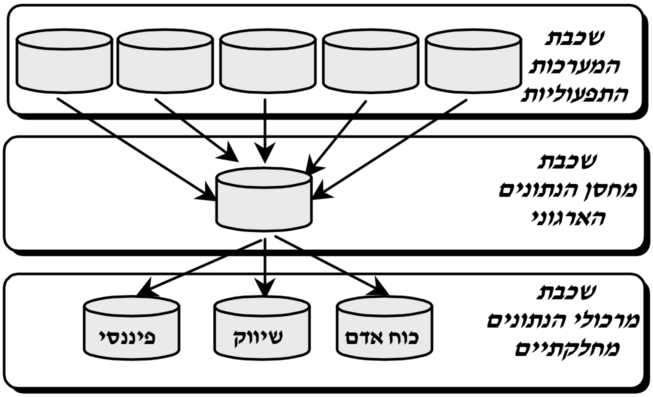
מרכול נתונים בארכיטקטורה תלת-שכבתית (Three Tier Data Mart Architecture)

מרכול נתונים זה מבוסס על מחסן הנתונים הארגוני, ולכן מוזן בנתונים ממנו, ולא באופן ישיר מהמערכות התפעוליות. ארכיטקטורה זו היא תלת-שכבתית, מפני שקיימות שכבת המערכות התפעוליות, שכבת מחסן הנתונים הארגוני ושכבת מרכולי הנתונים המחלקתיים. לעיתים מקובל לקרוא לסוג זה של מרכול נתונים **מרכול נתונים תלוי (Dependent Data Mart)**. מרכול הנתונים התלוי מהווה נגזרת של מחסן הנתונים הארגוני מפני שנתוניו נגזרים ממחסן נתונים ארגוני. מאחר והנתונים נגזרים ממחסן נתונים ארגוני, הם מבוססים על מודל נתונים גלובלי. ארכיטקטורה זו מפורטת בהמשך.



תרשים 11.3: מרכול נתונים בעל ארכיטקטורה תלת-שכבתית.

כפי שניתן לראות מהתרשים, כל השכבות הקיימות במחסן הארגוני ובמרכול הנתונים, קיימות גם כאן. מכיון שבארכיטקטורה זו קיימת אבחנה בין שכבת הנתונים התפעוליים לבין שכבת מחסן הנתונים הארגוני, ובין שכבת הנתונים המיועדים לתמיכה בקבלת החלטות, מקובל לכנות ארכיטקטורה זו **ארכיטקטורה תלת-שכבתית מבחינת הנתונים**.



תרשים 11.4: שכבות בארכיטקטורה תלת-שכבתית.

ארכיטקטורה זו מומלצת ונפוצה. היא משתמשת בתפיסת מחסן הנתונים הארגוני, המבוסס על מודל נתונים אחיד והיא יעילה בתהליכי הגזירה מהמערכות התפעוליות. ארכיטקטורה זו מאפשרת הקמת מרכולי נתונים ייעודיים, הבאים לפתור בעיות ביצועים, צרכים של כלי ניתוח ייעודיים וכדומה.

סוג בסיס הנתונים של מרכול הנתונים

בעוד שבמרכזו של מחסן הנתונים הארגוני ברור שעומד בסיס נתונים טבלאי, המסוגל לנהל היקפי נתונים גדולים באופן אמין ובביצועים טובים, הרי שבכל הקשור לבסיס הנתונים של מרכול הנתונים קיימות מספר חלופות - בסיס נתונים טבלאי, בסיס נתונים רב מימדי, שילוב כלשהו של שניהם או מבנה נתונים ייעודי. נסקור בקצרה כל אחת מחלופות אלו.

- **בסיס נתונים טבלאי:** גם מרכול נתונים יכול להיות מוקם על מערכת RDBMS מסחרית רגילת. המערכות המסחריות הנפוצות ביותר להקמת מרכולי נתונים הן למשל, מתוצרת Oracle, Informix, Sybase ו-DB/2 ו- SQL Server. האחרון, שבדרך כלל אינו משמש כבסיס נתונים של מחסן הנתונים הארגוני, נחשב לאחד מבסיסי הנתונים הנפוצים ביותר בסביבת מרכול הנתונים. בסיס הנתונים היחסי בסביבת מרכול הנתונים יכול להכיל גם מידע פרטני וגם מידע סיכומי, והוא משרת כלי שאילתות וניתוח מידע רגילים. עיצוב בסיס נתונים זה מבוסס גם הוא על תפיסות העיצוב המיוחדות לסביבות מחסני הנתונים - סכמות כוכב, סכמות פתיתי שלג וכדומה.

- **בסיס נתונים רב-מימדי:** חלק מכלי הניתוח הרב-מימדיים מבוססים על בסיס נתונים ייעודי רב-מימדי (MDB, קיצור של Multi Dimensional Database Management System). בסיס נתונים זה הוא בסיס נתונים נפרד, הנטען בנתונים הנגזרים ממחסן הנתונים, או ישירות ממערכות תפעוליות (הדבר אינו מומלץ). היתרון בבסיסי נתונים אלה הוא זמני התגובה המעולים שלהם. אלה נובעים, בין היתר, מביצוע חישובים שונים והכנת סיכומים מראש תוך כדי הטעינה. חלק מכלים אלה מאפשרים גישה שקופה לנתונים הפרטניים, המאוחסנים בבסיס הנתונים היחסי במחסן הנתונים.

- **מבנה נתונים ייעודי:** חלק מכלי כריית הנתונים זקוקים למבנה נתונים ייעודי. חלק מתהליך כריית הנתונים כרוך בגזירת הנתונים ממחסן הנתונים, או ממערכות אחרות, וביצוע הכנות מיוחדות של הנתונים. לדוגמה, קידוד הנתונים בצורה מיוחדת, הכנסת טווחים וכדומה. חלק מכלים אלה זקוק למבנה נתונים מיוחד לצורך הפעלה יעילה של האלגוריתמים לניתוח וכריית הנתונים.

שיקולים למימוש מרכול הנתונים בארכיטקטורה תלת-שכבתית

כפי שכבר נאמר בפרק זה, הדרך הנכונה להתייחס למרכול הנתונים היא כאל הרחבה טבעית של מחסן הנתונים לדרג המחלקתי. בהקשר זה, לתפיסת מרכול הנתונים יש חשיבות ומקום ברור במסגרת התפיסה הכוללת של מחסן הנתונים הארגוני. ארכיטקטורת מחסן נתונים ארגוני רב-שכבתי היא ארכיטקטורה נכונה ומתאימה. לכן, קיימים מצבים ברורים בהם מומלץ מאוד להשתמש בה. נסקור את הסיבות העיקריות לבחירה בתפיסת מרכול הנתונים, כחלק מהארכיטקטורה הכוללת של מחסן הנתונים הארגוני:

- **שיפור ביצועים:** לעיתים, צריך להריץ שאילתות ממוקדות מספר רב של פעמים, וזמני התגובה של שאילתות אלה הוא קריטי לתהליך קבלת ההחלטות. מכיון שהיקף הנתונים במחסן הנתונים הארגוני הוא גדול ואמור לשרת מספר רב של משתמשים שונים, אחת הדרכים להשיג ביצועים מעולים היא לגזור את הנתונים מתוכו, להעבירם לשרת ייעודי ולבצע את ניתוח המידע במרכול הנתונים הייעודי.
- **התאמת מבנה הנתונים לכלי ניתוח מידע:** מכיון שאין בנמצא כלי ניתוח מידע אחד המסוגל לבצע את כל סוגי ניתוח המידע, ולשמש את כל סוגי המשתמשים, תפיסת מחסן הנתונים הארגוני מעודדת ומכירה בצורך במספר רב של כלי ניתוח מידע, כל אחד מותאם לצרכים מיוחדים. לעיתים, כלי ניתוח אלה מבוססים על בסיסי נתונים ייעודיים (כמו, למשל, כלי ניתוח רב-מימדיים המבוססים על בסיס נתונים רב-מימדי או כלים מסוימים לכריית נתונים המבוססים על מבנה ייעודי).
- **אבטחת מידע:** לעיתים צריך לשקול את נושא אבטחת המידע במחסן הנתונים הארגוני. אחת הדרכים האפשריות היא גזירת נתונים ממחסן הנתונים הארגוני לשרת נפרד ובסיס נתונים נפרד, ולאפשר לחלק מהמשתמשים גישה לשרת זה בלבד.
- **פיצול טבעי של הנתונים:** קיימים מצבים בהם נתונים מסוימים אכן משמשים אך ורק מחלקה מסוימת, ואין בהם עניין למחלקות אחרות. במצבים כגון אלה אין הגיון להתעקש על הכנסת נתונים אלה למחסן הנתונים הארגוני. עדיף לנהל אותם במסגרת מרכול נתונים. לדוגמה, מחלקות שונות אחראיות על קווי מוצר שונים. ניתן להקים לכל מחלקה מרכול המנהל את הנתונים הרלוונטיים לקו המוצרים שלה, תוך גזירת הנתונים ממחסן הנתונים הארגוני.
- **פיילוט ושלב ביניים בתהליך הקמת מחסן הנתונים הארגוני:** לעיתים, יש לארגון ספקות לגבי תפיסת מחסן הנתונים הארגוני, והוא מעוניין לבצע בחינה ממוקדת של התפיסה. לכן, הוא משתמש בתפיסת מרכול הנתונים כפיילוט לתפיסת מחסן הנתונים, וכשלב ביניים לפני קבלת החלטה להיכנס לפרויקט שאפתני יותר להקמת מחסן נתונים ארגוני.

מומלץ בחום להקים מרכול נתונים אך ורק בשל אחד משיקולים אלה. לא בשל שיקולים אחרים, כגון מהירות ההקמה, עלות נמוכה, לחצים המופעלים על ידי מחלקה מסוימת וכדומה. בסופו של דבר, התבססות על תפיסת מרכול הנתונים משיקולים אחרים תביא לעלויות פיתוח ותחזוקה גבוהות, חוסר עקביות בנתונים המנוהלים במרכולים שונים, השקעה גבוהה בתהליכי גזירה מיותרים והתפתחות של איי מידע מבודדים.

מודלים לפיתוח והקמת מרכולי נתונים

במשך השנים התפתחו שלושה מודלים שונים לפיתוח ובניית מרכולי נתונים:

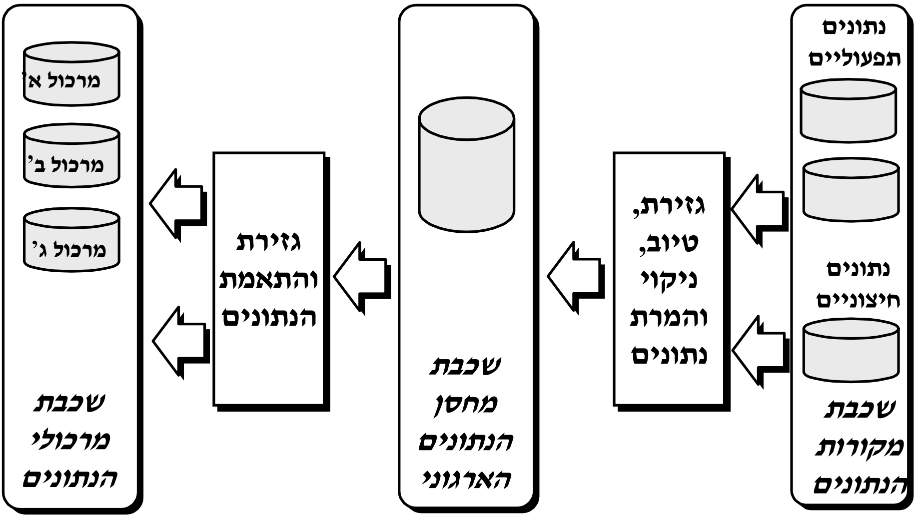
- **מודל פיתוח מעלה-מטה** - גישה זו מבוססת על ההבנה שמרכול הנתונים הוא הרחבה טבעית של מחסן הנתונים הארגוני, וככזה הוא מוזן בנתונים הנגזרים ממחסן הנתונים הארגוני.
- **מודל פיתוח מטה-מעלה** - גישה זו מתחילה בפיתוח מרכולי הנתונים בתקווה שבסופו של דבר הם ישמשו כבסיס למחסן הנתונים הארגוני.
- **מודל פיתוח מקבילי** - בגישה זו מפותחים מרכולי הנתונים במקביל לפיתוח המחסן הארגוני. פיתוח מרכולי הנתונים בגישה זו אינו מתעכב עד השלמת הקמתו של מחסן הנתונים הארגוני, אלא מתפתח במקביל לו, תוך התבססות והתחשבות בעקרונות ובארכיטקטורה שמכתיב מחסן הנתונים.

נפרט בקצרה כל אחת מגישות אלו.

פיתוח מרכול נתונים במודל מעלה-מטה (Top Down Model)

זהו מודל הפיתוח האלגנטי ביותר. בשלב ראשון מוקם מחסן הנתונים הארגוני, ובשלב שני מוקמים מרכולי הנתונים הרלוונטיים ועל פי צרכי המחלקות השונות. מרכול הנתונים מוזן בנתונים שמקורם במחסן הנתונים. אי לכך, אלה נתונים שעברו תהליכי טיוב, ניקוי והמרה, ולכן הם עקביים ומותאמים לתהליכי קבלת החלטות.

במודל פיתוח זה, מחסן הנתונים מהווה את נקודת האינטגרציה של הנתונים מכל מקורותיהם. בעיקרון, כל הנתונים הדרושים לתמיכה בתהליכי קבלת ההחלטות נמצאים ומאוחסנים במחסן הנתונים. עם סיום הקמתו מפותחים מרכולי נתונים, שמטרתם שיפור ביצועים, הבאת הנתונים קרוב יותר למחלקה המשתמשת בהם או התאמת הנתונים לכלי הניתוח השונים. זרימת הנתונים בין מחסן הנתונים למרכול הנתונים היא חד כיוונית בלבד, ומרכולי הנתונים מכילים רק נתונים הנגזרים ממחסן הנתונים. אם, תוך כדי תהליך הקמת מרכול נתונים, מתברר שנתונים הדרושים למחלקה מסוימת חסרים, מודל הנתונים של מחסן הנתונים מורחב, הנתונים החסרים מועברים אליו וממנו אל מרכול הנתונים.



תרשים 11.5: מודל פיתוח מעלה-מטה בהקמת מרכול נתונים.

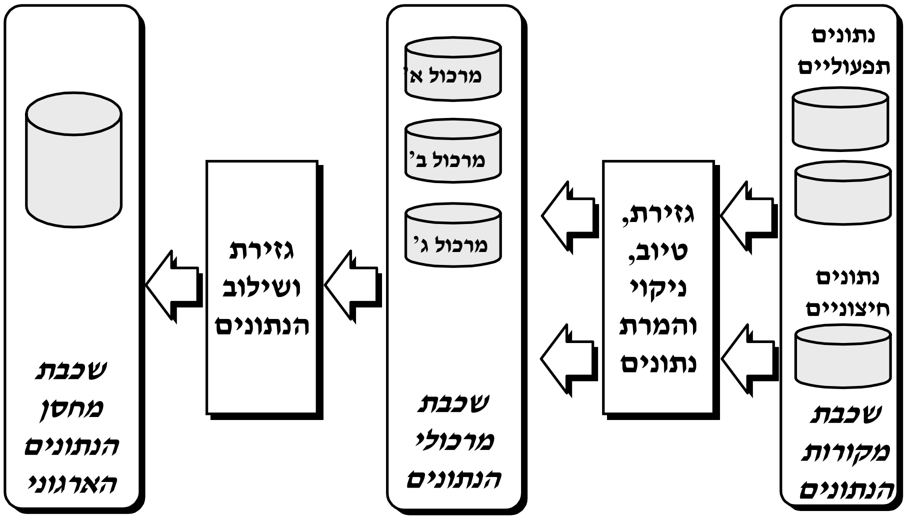
מודל זה מניח שלארגון יש את המשאבים והזמן להקים תחילה את מחסן הנתונים הארגוני, ורק לאחר מכן להתחיל בטעינת מרכולי הנתונים. הבעיה היא, שהנחה זו אינה תמיד תואמת למציאות, והלחצים למתן פתרונות מהירים לבעיות מקומיות, הם גדולים. במקרים כגון אלה, הזמן הנדרש להקמת המחסן הנתונים הארגוני אינו קביל.

פיתוח מרכול נתונים במודל מטה-מעלה (Bottom Up Model)

מודל פיתוח והקמה זה יוצא מנקודת הנחה כי ראשית מתבצעת הקמת מרכולי נתונים, ורק בשלב מאוחר יותר הקמת מחסן הנתונים הארגוני. לכן, נוצר מצב בו מחסן הנתונים מוזן, למעשה, בנתונים ממרכולי הנתונים. מרכולי הנתונים מוזנים ישירות בנתונים הנגזרים מהמערכות התפעוליות ובאמצעות תהליכי גזירה, מיטוב וניקוי.

כפי שניתן לראות, מודל הקמה זה יוצא מנקודת הנחה כי תחילה הוקמו מרכולי נתונים, כלומר איי מידע המיועד לתמיכה בקבלת החלטות. מרכולים אלה מבודדים, ומוקמים על ידי מחלקות שונות, על פי צרכיהן. כל אחד מהמרכולים דורש תהליכי גזירה מיוחדים ותהליכי טיוב, ניקוי והמרה ייעודיים. לכן, נוצרת כפילות במאמצים המושקעים. במובן מסוים, ניתן לומר שבתפיסה זו מקים הארגון מערכות Legacy חדשות, שבשלב כלשהו מאוחר יותר ייאלץ להתמודד עם שילובן והחלפתן. רק בשלב מאוחר יותר מתחיל הארגון בתהליך שילוב מרכולי הנתונים למחסן הנתונים הארגוני. תהליך השילוב אינו פשוט. הוא דורש ניתוח מדויק של תכולת הנתונים בכל מרכול, פתרון הסתירות בין ההגדרות, איחוד שיטות ייצוג שונות, סילוק כפילויות וכדומה.

שאלה חשובה במודל הקמה זה היא מה קורה לאחר שמחסן הנתונים מוקם. האם הוא יחליף את המרכולים? האם יזין אותם, או האם הוא ימשיך להיות מוזן מהם? בכל מקרה, חשוב שארגון יבין שתהליך בניית מחסן נתונים ממרכולי נתונים אינו פשוט כלל וכלל, וידרוש מאמץ ניכר. יתר על כן, הוא אף ידרוש שינוי בהרגלי העבודה שהתבססו כבר במחלקות השונות.

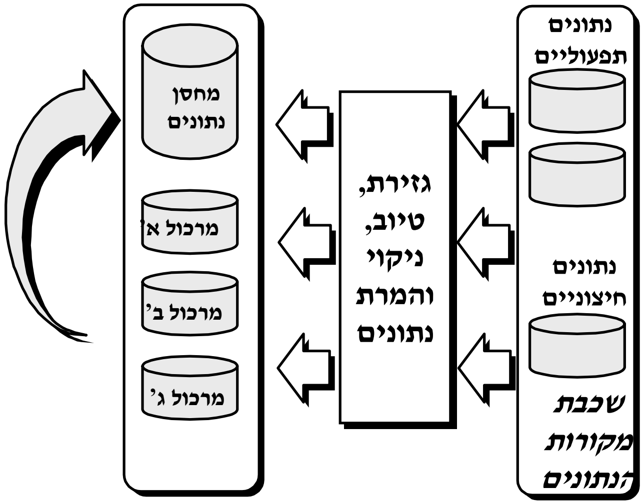


תרשים 11.6: מודל פיתוח מטה-מעלה בהקמת מרכול נתונים.

פיתוח מרכול נתונים במודל מקבילי (Parallel Model)

מודל ההקמה הנפוץ ביותר הוא מודל ההקמה המקבילי. מודל זה פותר, לפחות ברמה העקרונית, את הבעיות האינהרנטיות של כל אחד מהמודלים הקודמים. מצד אחד, מבוסס מודל זה על ההכרה שמרכול הנתונים חייב להיות מבוסס על מחסן הנתונים, על ארכיטקטורה אחידה ועל מודל נתונים מוסכם ואחיד. מצד שני, מודל זה יוצא מנקודת הנחה שהלחצים למתן פתרונות מהירים גדולים, ולכן חייבים לאפשר פתרונות מקומיים, בד בבד עם הקמת המחסן הארגוני.

תפיסה זו מניחה שמרכולי הנתונים ייבנו בהתבסס על מודל נתונים ארגוני, תוך פתרון סתירות, הגדרות מוסכמות ואחידות. כל כפילויות הנתונים ייטענו באופן מבוקר ויסולקו תוך כדי התקדמות בניית מחסן הנתונים. למחלקות ניתנת מידה מסוימת של עצמאות בהקמת מרכול הנתונים, ובקביעת הנושאים שינוהלו במסגרתו. עצמאות זו מלווה בבקרה מרכזית. לכן, שילובם בסופו של התהליך כחלק ממחסן הנתונים, יהיה קל יחסית. המודל מניח שתהליך הקמת המחסן הארגוני ייהנה מהלקחים שיילמדו בעת תהליך הקמת מרכולי הנתונים. עם סיום הקמת המחסן הארגוני, הוא יהפוך למקור הנתונים של מרכולי הנתונים. בשלב זה תיפסק הזנת מרכולי הנתונים בנתונים הנגזרים ישירות מהמערכות התפעוליות וממקורות אחרים.



תרשים 11.7: מודל פיתוח מקבילי בהקמת מרכול נתונים.

למרות שמודל זה מציאותי יותר, ולוקח בחשבון את מערכת הקשרים העתידית בין מחסן הנתונים ומרכולי הנתונים, המציאות לפעמים טופחת על פני המודל. המחלקות מפעילות לחצים לאספקה מהירה של פתרונות, ולא תמיד מוכנות להבין את חשיבות התכנון המסודר וההתבססות על ארכיטקטורה אחידה. מבחינתן, תהליך זה אורך זמן והן אינן מוכנות לקבל זאת. למרות קשיים אלה, ארגון שיהיה מוכן להשקיע את האנרגיה במימוש מודל הקמה זה, ירוויח בסופו של דבר ויהנה ממחסן נתונים ארגוני הפועל במשולב עם מרכולי נתונים מחלקתיים, המפותח בו זמנית במידה רבה של תאום ושל עצמאות.

סיכום

מרכול הנתונים מהווה הרחבה טבעית של תפיסת מחסן הנתונים הארגוני אל רמת המחלקה. מחסן הנתונים הארגוני מנהל את הנתונים ברמה הפרטנית. לעומתו, מרכול הנתונים יכול לנהל את הנתונים ברמות סיכומיות, בהתאם לדרישות המחלקה, או במבני נתונים ייחודיים הנדרשים עבור כלי הניתוח השונים.

הפיתוי במימוש תפיסת מרכול הנתונים המבודד ברור. קיצור לוחות זמנים, מחסני נתונים קטנים יותר (ולכן קלים יותר לעיצוב), בנייה ותפעול, יכולת המחלקות השונות לקבוע לעצמן את סדרי העדיפויות והמשאבים (מבלי להיות תלויים בשקולים ומשאבים של מחלקות אחרות). חלק מהסיבות שארגונים מיהרו לאמץ תפיסה זו הן שמספר גדול של ספקים וחברות ייעוץ מציעות לארגונים חבילות שלמות הכוללות ייעוץ, תוכנה, חומרה והדרכה, במחירים זולים יחסית ותוך הבטחה להקים מרכול נתונים בפרקי זמן קצרים ביותר (מספר חודשים מצומצם). לדוגמה, באחד הירחונים הופיע מודעת ענק: **אנו נקים עבורך מרכול נתונים תוך חודש**. איך בכלל ניתן לעמוד בפיתוי זה למימוש רעיון טוב תוך פרק זמן קצר ובעלויות סבירות? ארגונים מסוימים, בשל גודלם, מבוזרים מאוד

ומשרתים לקוחות ומוצרים שונים. בארגונים כאלה ייתכן שאין דרך אחרת, וכל ניסיון להקים מחסן נתונים ארגוני נועד מראש לכישלון. לכן, הדרך הישימה היחידה היא בחירה בארכיטקטורת מרכול הנתונים.

העלות הנמוכה וקלות התפעול של שרתים מרובי מעבדים, המסוגלים לנהל בסיסי נתונים של מספר עשרות ג׳יגה בתים, הביאה לכך שפופולריות מרכולי הנתונים המבודדים הרקיעה שחקים. אין צורך להתמודד עם בעיות תקציביות קשות, ולכן קל לשכנע את ההנהלה לאמץ ארכיטקטורה זו.

הסכנה הגדולה ביותר בתפיסת מרכול הנתונים טמונה דווקא בהצלחתם. מחלקה שהצליחה ביישום מרכול נתונים מחלקתי אחד, מעודדת את המחלקה הבאה וכך הלאה. נוצר מצב של ״שיטפון״ במרכולי נתונים מחלקתיים. ארגון שאינו מוכן להשקיע בתכנון מראש של ארכיטקטורה זו ולנתח ולהבין את מלוא המשמעויות שלה, מסתכן בבניית סביבה לאספקת מידע ניהולי, שאולי תפתור בעיות מסוימות, אבל יחד עם זאת גם תביא איתה גל חדש של בעיות וכאב ראש לא קטן. הבעיות תורגשנה בפיתוח ותחזוקת תהליכי גזירה מהמערכות התפעוליות, בתחזוקת מספר רב של בסיסי נתונים (לעיתים אפילו של יצרנים שונים) המבוססים על מודלים סותרים, ומכילים רמה גבוהה של כפילות נתונים.

תפיסת מרכול הנתונים מחדדת עוד יותר את הרעיון Think Big - Start Small, **חשוב בגדול אך התחל בקטן**, ולפעמים גם במקביל. משמעות הדבר היא שיש להשקיע מאמץ בתחילת הדרך, להגדרת הסטנדרטים והארכיטקטורה (כלומר, הגדרת התמונה הכוללת, עקרונות וכלים לגזירת הנתונים, דרך יצירת תיאום מסוים בין מרכולי הנתונים השונים, ושיטות הבקרה). לאחר מכן, ניתן להתחיל במימוש מרכולי נתונים אינטגרטיביים, כל אחד בקצב שלו, עם המשאבים שלו ועם העדיפויות שלו, אבל כולם תחת המטריה הכוללת של מספר נושאים משותפים.

פרק 12: מתודולוגיה ליישום מחסן נתונים

מבוא

פרק זה עוסק באחד הנושאים המרכזיים בתפיסת מחסן הנתונים: כיצד מפתחים ומיישמים מחסן נתונים, ובמילים אחרות: **מהי המתודולוגיה ליישום**. מערכות מחסני נתונים שונות ממערכות מידע תפעוליות לא רק במטרות, בקהל המשתמשים ובמבנה הפנימי שלהן, אלא גם בצורת הקמתן. תהיה זו טעות להניח שקיים דמיון רב בין פיתוח ויישום מערכת מידע תפעולית חדשה לבין פיתוח ויישום מחסן נתונים. המציאות היא שקיים שוני מהותי בין מחזור החיים ומתודולוגיית היישום של שני סוגי מערכות אלו. פרק זה יציג את מחזור החיים של מערכת מחסן נתונים ואת המתודולוגיה ליישומה.

כמו בכל מתודולוגיה, גם המתודולוגיה ליישום מחסני נתונים מחלקת את מחזור החיים של המערכת למספר שלבים מוגדרים, ומציעה אוסף מובנה של פעולות וצעדים שיש לבצע מתחילת הפרויקט ועד ליישום המלא. שימוש ב"ספר בישול" כזה, המבוסס על ניסיון מצטבר במספר רב של יישומים בארגונים שונים, מקטין את הסיכוי לכישלון הפרויקט ומבטיח רמת איכות גבוהה של התוצרים השונים שיופקו במשך תהליך היישום.

פרויקט ההקמה של מערכות מחסני נתונים ארגוניים יכול להימשך בין שנה לשנתיים, על פי היקפו. זהו פרויקט מורכב בעל מספר אפיונים חשובים:

- **שילוב מספר רב של משאבים שונים:** פרויקט מחסן נתונים כולל מספר רב של משאבים שונים וביניהם משתמשים שונים, מנתחי מערכות, מפתחי יישומי מחסן נתונים, מנהלי בסיסי נתונים, מפתחי תהליכי גזירה וניקוי, יועצים חיצוניים, ספקים שונים וכדומה.
- **שילוב מספר רב של טכנולוגיות מחשוב:** פרויקט מחסן נתונים מבוסס על טכנולוגיות מחשוב מודרניות שונות. טכנולוגיות אלו כוללות, בין השאר: שרתים רבי עוצמה עם נפחי אחסון גדולים מאוד, מערכות RDBMS המשתמשות בטכניקות עבודה מקבילית בסביבה מרובת המעבדים, כלי אחזור רבים ומתוחכמים, כלי תוכנה לגזירת והעברת נתונים בין פלטפורמות מחשוב שונות, רשתות תקשורת מחשבים מהירות, אינטראנט להפצת השאילתות בין המשתמשים השונים וכדומה.

- **משך זמן יישום ארוך יחסית:** פרויקט מחסן נתונים ארגוני הינו תהליך ארוך, שעשוי להימשך בין שנה לשנתיים.
- **עלות גבוהה:** השילוב של מספר רב של משאבים, של טכנולוגיות מחשוב ומשך זמן יישום ארוך יחסית גורם לכך שפרויקט הקמת מחסן נתונים הינו פרויקט יקר יחסית.

מכל האמור לעיל ניתן ללמוד שארגון המחליט ליישם מערכת מחסן נתונים חייב להתבסס על מתודולוגיה כלשהי שתבטיח את ביצוע הפרויקט והצלחתו. במשך השנים התפתחו מספר מתודולוגיות יישום שונות, שלכל אחת מהן דגשים שונים אך עם זאת, יש דמיון רב ביניהן.

המתודולוגיה המוצגת בפרק זה מבוססת בחלקה על המתודולוגיה Hadden-Kelly Data Warehouse Method שפותחה על ידי שני יועצים מובילים בתחום, ושהיתה לי ההזדמנות ללמוד אותה מפיו של אחד מהם, Earl Hadden, באחד הסמינרים שהעביר בשנת 1996 בלונדון.

עקרונות המתודולוגיה

מחזור החיים לבניית מחסן נתונים מבוסס על שני עקרונות יסוד: בנייה הדרגתית ושיפור מתמיד.

- **בנייה הדרגתית (Incremental Approach):** מחזור החיים מניח שבניית מחסן נתונים ארגוני הינה משימה גדולה ומורכבת. על כן, הגישה הנכונה היא בנייה הדרגתית, כך שבכל פעם נבנה מקטע נוסף של מחסן הנתונים. המקטעים נבנים בצורה מתואמת ובהתבסס על אסטרטגיה כוללת המגובשת בשלב הראשון.
- **שיפור מתמיד (Constant Enhancement):** את מחסן הנתונים מאפיין פיתוח מתמיד, ולכן נדרש מהלך מתמיד של שיפור והענות לדרישות חדשות שלא ניתן היה לצפות בשלבים מוקדמים. קצב השינויים במחסן הנתונים גבוה בהרבה ממה שאנו רגילים לו בפיתוח מערכות מידע רגילות ומחזור החיים חייב לקחת עובדה זו בחשבון.

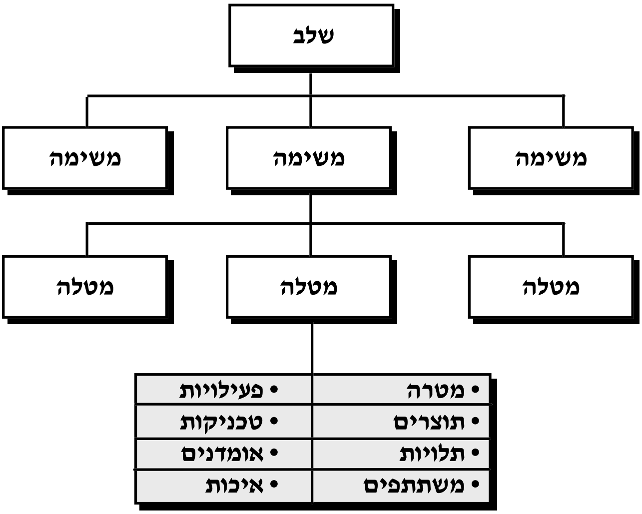
תהליך הקמת מחסן נתונים נמשך זמן וצורך משאבים רבים, ולכן יש קושי אמיתי להצדיק כלכלית פרויקט שמפרותיו ניתן ליהנות רק כעבור זמן רב כל כך. על רקע קושי זה ועל רקע כישלונות רבים של ארגונים בהקמת מחסן נתונים, מחולק פרויקט ההקמה והיישום למספר שלבים ברורים, ברי ניהול ושליטה, שאפשר להפיק מהם תוצרי ביניים במהלך הפרויקט. כפי שנראה בהמשך, המתודולוגיה המוצגת בפרק זה מבוססת על תהליך תוספתי (אינקרמנטלי), המחלק את מחסן הנתונים למספר מקטעים שונים. כך, בכל איטרציה מקטע מסוים של מחסן הנתונים מתחיל לפעול והארגון יכול ליהנות בשלב מוקדם יחסית מתוצרי המערכת. במובן זה, המתודולוגיה מביאה את הארגון אל התוצר הסופי באופן הדרגתי. נדגיש שהתפיסה התוספתית העומדת בבסיס המתודולוגיה אינה מקצרת את משך הפרויקט - שנה עד שנתיים - אלא מאפשרת לארגון ליהנות מתוצרי

מחסן הנתונים בשלבים מוקדמים יותר ולבסס את השלבים הבאים על לקחים שהופקו מיישום השלבים הקודמים. שיטה זו מבטיחה בקרה טובה יותר על הפרויקט. אם יתרונות מסוימים לא מושגים, או שנתקלים בקשיים לא צפויים, עדיין נותר בידי הארגון זמן לתקן את הדורש תיקון, ולבחון את הפרויקט וכדאיותו עוד לפני ביצוע השקעות.

נקודה נוספת שיש להדגיש היא שהמתודולוגיה המוצגת מיועדת להקמת מחסן נתונים ארגוני, ולא מרכול נתונים, אך ניתן לגזור רעיונות ממתודולוגיה זו עבור הקמת מרכולי נתונים. עם זאת, הדגש העיקרי במתודולוגיה נשאר בעינו: הקמת מחסן נתונים ארגוני החוצה את גבולות האגפים והמחלקות השונות ובעל תכולת מידע במיגוון רחב של נושאים המעניינים את כלל הארגון. אין כל ספק שהיתרונות העסקיים והתועלות שמחסן נתונים ארגוני מציע הם רבים. עם זאת, אסור לשכוח שתהליך ההקמה מורכב ומשך ההקמה ארוך.

מבנה המתודולוגיה

המתודולוגיה להקמת מחסני נתונים מחלקת את מחזור החיים של תהליך הקמת מחסן הנתונים למבנה היררכי המורכב ממספר שלבים, כאשר כל שלב מורכב ממספר משימות וכל משימה מורכבת ממספר מטלות. התרשים הבא מציג את המבנה העקרוני של המתודולוגיה.



תרשים 12.1: מבנה עקרוני של המתודולוגיה.

המתודולוגיה מחלקת את מחזור החיים של מחסן הנתונים לארבעה שלבים:

- ❖ בחינת הצורך העסקי והמוכנות הארגונית,
- ❖ גיבוש ארכיטקטורת-על,
- ❖ בניית מקטע של מחסן הנתונים,
- ❖ תפעול מקטע.

המתודולוגיה היא תוספתית במהותה, ולכן שני השלבים הראשוניים הם חד-פעמיים בשעה ששני השלבים הבאים חוזרים על עצמם מספר פעמים, עבור כל מקטע. כל אחד מהשלבים מורכב ממספר משימות שכל אחת מהן מורכבת ממספר מטלות. עבור כל מטלה מתארת המתודולוגיה את הנושאים הבאים:

- **מטרה (Objective)** - תיאור קצר של המטרה בביצוע המטלה.
- **תוצרים (Deliverables)** - התוצרים העיקריים של המטלה, כולל התייחסות לשימוש שנעשה בהם.
- **תלויות (Dependencies)** - ציון משימות ומטלות שיש לסיים לפני תחילת ביצוע המטלה הנוכחית, אילו מטלות יכולות להתבצע במקביל למטלה זו וכדומה.
- **משתתפים (Participants)** - זיהוי המשתתפים בכל מטלה והגדרת האחריות של כל משתתף.
- **פעילויות (Activities)** - אוסף כל הפעילויות שיש לבצע במסגרת המטלה.
- **טכניקות (Techniques)** - תיאור הטכניקות והכלים שיש להשתמש בהם לביצוע המטלה (כמו לדוגמה, תרשימי ERD, תרשימי סכמות כוכב וכדומה).
- **אומדנים (Estimates)** - הערכת הזמן שיש להקצות לביצוע המטלה, ובכלל זה התייחסות לסיכונים שיכולים לגרום לסטייה מהתכנון.
- **איכות (Quality)** - מדדים והנחיות שיאפשרו את השגת האיכות ואת הצלחת המטלה.

תיאור מלא של כל הנושאים שיש לבצע בכל אחת מהמטלות היא מעבר למטרות ספר זה. בהמשך נציג תיאור עקרוני של השלבים והמשימות. המעוניין בפרטי המתודולוגיה יוכל לפנות אל המפתחים.

השלבים במחזור החיים של מחסן הנתונים (Data Warehouse Life Cycle)

התרשים הבא מציג את השלבים העיקריים במחזור החיים של מחסן הנתונים. נסקור בקצרה כל אחד משלבים אלה.

- **שלב 1 - בחינת הצורך העסקי ומוכנות הארגון למחסן נתונים (Business Need and Organizational Readiness):** השלב הראשון הוא בדיקת מוכנות הארגון לפרויקט מחסן הנתונים. מחסן הנתונים הארגוני הינו פרויקט בעל חשיבות אסטרטגית לארגון, יקר יחסית למימוש והקמתו נמשכת זמן רב. על כן דרושה בחינת המוכנות הארגונית ורמת המחויבות של הנהלת הארגון לנושא. במסגרת שלב זה מקובל לבחון את יעדי העל של הפרויקט ואת הצרכים העסקיים שמחסן הנתונים צריך לענות עליהם.

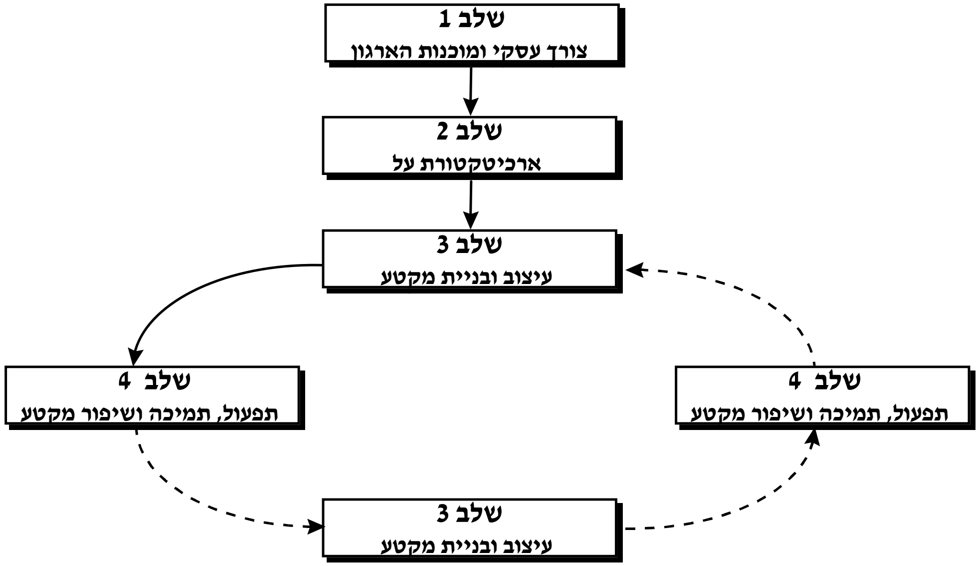
❖ **שלב 2 - גיבוש ארכיטקטורת העל (Data Warehouse Architecture):** במסגרת שלב זה נקבעת אסטרטגיית המימוש והארכיטקטורה הכוללת של מחסן הנתונים. אחת ההחלטות האסטרטגיות העיקריות שיש לקבל בשלב זה מתייחסת לשיטת מימוש מחסן הנתונים: גישת מעלה-מטה או מטה-מעלה. שלב זה הוא בעל חשיבות מרובה, כי הוא מהווה את תוכנית האב הקובעת את הנושאים, הסטנדרטים והתשתיות לכל שאר השלבים. במסגרת שלב זה מגדירים את המקטעים השונים של מחסן הנתונים, את העדיפויות ואת סדר ההקמה שלהם. שלב זה קצר יחסית למשך הפרויקט, בין חודש לשלושה חודשים, על פי גודל הארגון והיקף מחסן הנתונים העתידי. הקצב הגבוה יחסית של שינויים ארגוניים שיש להם השלכה על מחסן הנתונים מחייב לעדכן מעת לעת את מסמך ארכיטקטורת העל, כדי שישקף ככל הניתן את המציאות.

❖ **שלב 3 - עיצוב ויישום מקטע (Data Warehouse Increment Design and Implementation):** בשלב זה מעצבים מקטע מסוים של מחסן הנתונים ומאחסנים בו נתונים. המקטע מוגדר כנגזרת חלקית מסוימת מתוך המחסן הכולל (לדוגמה, נושא מסוים, חלק מנושא רחב יותר וכדומה). בגלל ההיקף הגדול של מחסן הנתונים הארגוני, כל ניסיון להקימו בבת אחת הוא מסוכן ובדרך כלל נועד לכישלון. כדי להימנע מבעיות אלו, מחלקים את המחסן למספר מקטעים הנבנים זה אחר זה תוך ביסוס כל מקטע על המקטעים שלפניו. משך היישום של מקטע צריך להיות בגבולות זמן של שלושה עד שישה חודשים. עיקרון מנחה זה עומד בבסיס התפיסה התוספתית המבוססת על שחרור הדרגתי של מחסן הנתונים לשימוש. מקובל לקרוא למקטע הראשון מקטע הבסיס (Baseline) כי במקטע זה הארגון מתנסה לראשונה עם תפעול מחסן נתונים. סביר להניח שהלקחים שיופקו בהפעלת מקטע זה יהיו רבים יותר מאשר במקטעים הבאים.

❖ **שלב 4 - תפעול, תמיכה ושיפור (Data Warehouse Operation, Support and Ongoing Improvement):** בגמר פיתוח מקטע מסוים, הוא נכנס לשלב התפעול השוטף, תוך תמיכה שוטפת במשתמשים. ההנחה הבסיסית היא שמחסן הנתונים הינו דינמי וקשה לחזות מראש את דרישות המידע, ולכן יש להמשיך בתהליכי שיפור ביצועים, הפקת לקחים, הענות לדרישות חדשות, הכנסת כלי אוטומציה וכדומה. הלקחים הנלמדים בכל מקטע משמשים הן לשיפור המקטע עצמו והן לשיפור המקטעים הבאים שיופעלו, ובמידת הצורך - גם שיפור מקטעים קודמים שפועלים.

כפי שניתן לראות מתרשים 12.2, שלבים 3 ו- 4 חוזרים על עצמם בהקמה של כל מקטע חדש במחסן הנתונים. נקודה שאינה באה לידי ביטוי בתרשים, אולם כדאי להזכיר אותה היא, שתוך כדי מימוש המקטעים רצוי מאוד לעדכן מעת לעת את מסמך הארכיטקטורה. כך הוא ימשיך להיות רלוונטי וישקף את המצב האמיתי, ולא רק את הארכיטקטורה בתחילת הפרויקט. מסמך מעודכן כזה ימשיך להיות נקודת ייחוס חשובה לאורך זמן, גם לאחר שחלק מהאנשים שהשתתפו בתהליך ההקמה יעברו למשימות ולתפקידים אחרים.

בהמשך נציג סקירה נרחבת יותר של כל אחד מהשלבים המרכיבים את מחזור החיים.



תרשים 12.2: השלבים העיקריים במחזור החיים של מחסן נתונים.

נושאים לטיפול בכל שלב במחזור החיים

עד עכשיו הצגנו מימד אחד של מחזור החיים - את **השלבים**. נציג עכשיו מימד נוסף והוא **אוסף הנושאים** שיש לטפל ולהגדיר בכל אחד משלבי מחזור החיים. ניתן לחלק את אוסף הנושאים לארבע קטגוריות ראשיות: פונקציונליות, נתונים, טכנולוגיה ותמיכה. אשכולות נושאיים אלה עוברים כחוט השני דרך כל השלבים, כאשר בכל שלב נכנסים לרמת פירוט גבוהה יותר. נסקור בקצרה כל אחת מהקטגוריות.



תרשים 12.3: שלבים במחזור חיים, קטגוריות נושאים לטיפול ותוצרים.

- **פונקציונליות (Functionality):** קטגוריה זו עוסקת במכלול נושאים הקשורים להגדרת הפונקציונליות של מחסן הנתונים. במסגרת קטגוריה זו נגדיר את סוגי שאילתות ואת ניתוחי מידע שמחסן הנתונים צריך לתמוך בהם; נחליט האם השאילתות תפעלנה באופן ישיר מול מחסן הנתונים, או שנבנה על מרכולי נתונים ייעודיים; נזהה את סוגי המשתמשים השונים של מחסן הנתונים; ונקבע לאיזה סוגים של כלי אחזור וניתוח מידע הם זקוקים.
- **נתונים (Data):** קטגוריה זו עוסקת במכלול הנושאים הקשורים לאחד המרכיבים החשובים ביותר - קביעת הנתונים שינוהלו במחסן הנתונים. במסגרת קטגוריה זו נגדיר את הנתונים הדרושים למשתמשים, נעצב את מודל הנתונים של מחסן הנתונים, נמפה את מקורות הנתונים במערכות התפעוליות ובמערכות חיצוניות, נגדיר את תהליכי הגזירה, הניקוי והטיוב של הנתונים.
- **טכנולוגיה (Technological Infrastructure):** קטגוריה זו עוסקת במכלול נושאים הקשורים לתשתית טכנולוגיית המחשוב הדרושה כדי להפעיל ולתמוך במחסן הנתונים. במסגרת קטגוריה זו נגדיר את מערכת RDBMS הדרושה, נקבע את סוגי ועוצמת השרתים, נגדיר את נפחי דיסקים נדרשים, נבחן ונגדיר את תשתית תקשורת הנתונים הנדרשת לקישור סביבת מחסן הנתונים עם הסביבות התפעוליות ועם המשתמשים, נגדיר אם נדרשים כלים מיוחדים לגזירה והעברת נתונים וכדומה.
- **תמיכה (Support):** קטגוריה זו עוסקת במכלול נושאים הקשורים בתפעול מחסן הנתונים. במסגרת קטגוריה זו נגדיר איזה משאבי פיתוח נדרשים לתמיכה בתהליכי גזירת הנתונים והניקוי שלהם, נגדיר כמה מנהלי בסיסי נתונים (DBA) נדרשים, נגדיר כמה מנתחי מערכות נדרשים כדי לתמוך במשתמשים השונים ובכלי האחזור השונים.

הטיפול בכל אחת מקטגוריות אלו הוא הדרגתי. כלומר, נכנסים לרמת פירוט גבוהה יותר תוך כדי ההתקדמות בשלבי מחזור החיים. כדי להדגים את רמת הפירוט ההולכת וגדלה, נתבונן לדוגמה בקטגוריה של הנתונים, ונראה כיצד נושא זה מתפתח מהגדרה כללית מאוד בשלב גיבוש ארכיטקטורת העל, ועד להגדרה מדויקת מאוד בשלבים הבאים.

- בשלב **גיבוש ארכיטקטורת העל** נזהה רק את הנתונים העיקריים הנדרשים לתמיכה בנושאים שהוגדרו עבור מחסן הנתונים, נזהה את המקורות העיקריים שלהם (מערכות תפעוליות או חיצוניות שיכולות לספק את הנתונים) ונגדיר מודל נתונים ראשוני של מחסן הנתונים.
- בשלב **עיצוב ובניית המקטע** נפרט באופן מדויק את הנתונים הדרושים, נזהה בדיוק מה הם מקורות הנתונים מבחינת המערכות התפעוליות ובאילו פלטפורמות מערכות הן פועלות, מה איכות הנתונים במערכות אלו, נזהה אילוצים מיוחדים פועלים עליהם, נגדיר כיצד ניתן להביא את הנתונים אל מחסן הנתונים ונעצב את מודל הנתונים המדויק של מחסן הנתונים.

❖ בשלב **התפעול, התמיכה והשיפור** נוודא שאכן כל התהליכים לגזירת הנתונים מהמערכות התפעוליות פועלים כהלכה, נוודא שתהליכי אספקת הנתונים למשתמשים פועלים היטב, נבחן אפשרות להכנסת כלים אוטומטיים לשיפור תהליכי התפעול וכדומה. בנוסף, נזהה מהם הנתונים החדשים שיש להביא אל מחסן הנתונים כתוצאה משינוי בדרישות, מהיכן ניתן להביא אותם, כיצד יש לעדכן את מודל הנתונים של מחסן הנתונים על מנת לשקף את הנתונים החדשים וכדומה.

עד כאן סקרנו בקצרה את המבנה העקרוני של המתודולוגיה והשלבים השונים במחזור החיים של פרויקט מחסן נתונים. בסעיפים הבאים נסקור באופן מפורט יותר כל אחד מהשלבים ונציג מהם הנושאים לטיפול לפי ארבע הקטגוריות העיקריות ונפרט את התוצרים העיקריים.

פירוט השלבים במתודולוגיה

שלב 1: בחינת הצורך העסקי ומוכנות הארגון

מהות השלב

כל ארגון המבקש לבנות מחסן הנתונים חייב להסתכל על הנושא כפרויקט אסטרטגי של הארגון, בעל השלכות מרחיקות לכת על יכולתו לשרוד בעולם התחרותי. מאחר וזוהי החלטה אסטרטגית, יש לגזור אותה מתוך האסטרטגיה הכוללת של הארגון ושל טכנולוגיית המידע הארגונית. שלב זה הוא השלב הראשון בפרויקט ועוסק על כן בבחינת הצורך העסקי במחסן נתונים וברמת המוכנות הארגונית לקליטת מחסן הנתונים בארגון.

כמו שארגון בוחן כל השקעה מול הצורך והתועלות הצפויות לו, כך יש להתייחס גם להשקעה בהקמת מחסן הנתונים. בשלב זה מתבצעת בחינה של התועלות העסקיות שיכולות לנבוע מיישום מחסן הנתונים. כדי לזהות את התועלות העסקיות יש לבחון תחילה את התהליכים העסקיים העיקריים של הארגון, לחקור את בעיות המידע העיקריות בתהליכים אלה. יש לראות איזה מהתהליכים יטופלו במסגרת מחסן הנתונים, תוך התמקדות בתועלות העיקריות הצפויות מהמידע שיעמוד לרשות מקבלי ההחלטות.

הקושי, שאינו ייחודי להקמת מחסן נתונים, הוא חוסר היכולת שלנו לכמת את התועלות ולהפוך אותן למונחי כסף. קשה לכמת את התועלת במונחים כגון אלה: חיסכון בהוצאות שיווק של שני מיליון ש״ח, הגדלת מחזור המכירות ב-10 מיליון ש״ח כתוצאה מהגדלת נתח השוק בעוד 2 אחוז וכדומה. למרות הקושי הזה, חשוב להתמודד עם האתגר ולהגדיר את היעדים העיקריים שהארגון מקווה להשיג באמצעות יישום מחסן הנתונים. הנה לדוגמה:

❖ הגדלת נתח השוק ב- 20% תוך שנתיים.

❖ הפיכת 10% מהלקוחות הפוטנציאליים ללקוחות בפועל, כתוצאה ממיקוד מאמצי השיווק מול פלחי אוכלוסייה מסוימים.

- חיסכון של 50% בעלויות כוח אדם כתוצאה מיכולת וויסות טובה יותר של עובדים מול המטלות.

הקושי במדידת התועלת הצפויה, ולאחר מכן - במדידת השגת היעדים, אינו צריך לגרום למצב של "שב ואל תעשה". פרויקט מחסן נתונים שאין לו יעדים ברורים הנובעים באופן ישיר מאסטרטגיית הארגון צפוי לקשיים בקבלת המחויבות של ההנהלה למשאבים הנדרשים, ובסופו של דבר - רמת הסיכון בכישלונו גבוהה יותר.

זיהוי התהליכים העסקיים העיקריים (Business Processes Identification)

על מנת לאפשר יישום מחסן נתונים שיתפתח עם הזמן ויתמוך בתהליכי קבלת ההחלטות של הארגון, יש להקדיש זמן ומאמץ בלימוד התהליכים העסקיים העיקריים המעניינים את הארגון. יש להפריד בין התהליכים התפעוליים הרגילים של הארגון לבין התהליכים העסקיים הטקטיים והאסטרטגיים שלו, על אף שיש אינטראקציה ביניהם.

מחסן הנתונים מאורגן לפי נושאי מידע שכל אחד מהם תומך בתהליך עסקי אחד או יותר. כדי לזהות את נושאי המידע העיקריים שייושמו במחסן הנתונים, יש לזהות את הפעילויות ואת התהליכים העסקיים העיקריים מנקודת המבט של הארגון. נציג במספר דוגמאות של תהליכים עסקיים של סוגי ארגונים שונים:

- ניתוח רווחיות (Profitability Analysis) של לקוחות עסקיים ופרטיים בבנק.
- פילוח אוכלוסיות (Customer Segmentation) של לקוחות חברת טלקומוניקציה.
- איתור גניבות ושימוש חריג (Fraud Detection) במערכת הטלפונים.
- ניתוח עלויות המוצרים (Cost Analysis) בחברה תעשייתית.
- ניתוח תקלות בקווי הייצור.
- ניתוח מגמות כוח אדם בחברה מבחינת עיסוקים, השכלה וכדומה.

בשלב זה ייבדקו התהליכים העסקיים הקיימים והתהליכים העסקיים העתידיים, ככל שניתן לצפותם בשלב זה. לדוגמה, אם ידוע שהארגון צופה תחרות מתגברת ומתכוון לערוך שינויים במבנה הארגוני כדי להציע שירותים נוספים לפלחי לקוחות שונים, כמו עסקים גדולים, עסקים בינוניים, לקוחות ביתיים ואחרים.

על ידי הבנת התהליכים העסקיים המעניינים את הארגון, על ידי זיהוי הדרישות לטווח הקצר ועל ידי זיהוי הדרישות לטווח הארוך, נוכל לבנות מחסן נתונים שיסתגל עם הזמן לשינויים שיחולו בדרישות הארגון. רצוי מאוד להשקיע חלק מהמאמץ בשלב הזה ללימוד הצרכים העתידיים ולא להתמקד רק בטווח הקצר.

מומלץ לסווג כל אחד מהתהליכים העסקיים שיזוהו בשלב זה:

- **נושא.** כל תהליך עסקי ישויך לנושא מוגדר, על פי קבוצות לוגיות של שירותים. לדוגמה, תהליך ניתוח רווחיות של לקוחות, של סניפים ושל מוצרים שייכים כולם לקבוצת ניתוחי רווחיות.
- **קריטיות.** לכל תהליך תוגדר רמת הקריטיות שלו. תהליך קריטי הוא תהליך המשמש את מקבלי ההחלטות בעבודתם השוטפת. לכן, זמני התגובה, האמינות והזמינות שלו חייבים להיות גבוהים במיוחד.
- **ישימות.** ייבחנו הקושי או הפשטות ליישם את התהליך במסגרת מחסן הנתונים. הקושי יכול לנבוע מהעדר הנתונים המתאימים, או מתהליכי ניתוח מורכבים מאוד.
- **סוגי משתמשים.** לכל תהליך יזוהו המשתמשים העיקריים שלו. כפי שנראה בהמשך, מחסן הנתונים נבנה מתוך הנחה שהוא ישרת מיגוון רחב של סוגי אוכלוסיות משתמשים שונות.
- **ניתוח תועלות.** זיהוי התועלות העסקיות הצפויות כתוצאה מתמיכת מחסן הנתונים בתהליך. לנושא זה חשיבות רבה, כדי לאתר את התהליכים בעלי יחס תועלת/עלות הגבוהים ביותר. למרות הקושי הקיים בדרך כלל בהגדרת התועלות באופן כמותי, יש לעשות מאמץ ולתת לפחות הערכות כמותיות מסוימות לתועלות הצפויות.

זיהוי סדרי עדיפויות ליישום (Implementation Priorities)

בשלב זה ייקבעו הקדימויות ליישום על פי סדרי הקדימויות שהארגון קבע עבור התהליכים העסקיים ובהתאם ליישימות הלוגית והטכנולוגית. כלומר, ייבחנו ההיגיון ההנדסי בבניית מחסן הנתונים בהתבסס על רובדי הנתונים השונים שינוהלו בו. כלומר, יש לבחון לאיזה מרכיבים דרושים מרכיבים מקדימים לשם פעולתם התקינה.

הערכה ראשונית לגודל מחסן הנתונים

אחת השאלות הראשונות שעולות בתחילת יישום תפיסת מחסן הנתונים היא ההיקף הצפוי של הנתונים שינוהלו בו. לשאלה זו השפעה רבה על מורכבות הפרויקט, עלותו ומשך הקמתו, ולכן חשוב להגיע לאומדן ראשוני מוקדם ככל שאפשר. הטכנולוגיה העומדת בבסיס מחסן הנתונים הולכת ומשתכללת כל הזמן: עיבוד מקבילי, טעינה מהירה, חלוקה למחיצות, אינדקסים חדשים ומהירים, מערכות RDBMS המכירות את המבנה הייחודי של מחסן הנתונים ומבצעות אופטימיזציה טובה יותר של השאילתות, כלי גישה חכמים יותר. התפתחויות אלו תורמות ליכולת תמיכה ההולכת וגדלה וליכולת לנהל היקפי נתונים גדולים.

ארגון הנכנס לפרויקט מחסן נתונים צריך להעריך את היקף הנתונים שהוא עומד לנהל ואת השפעת גודל מחסן הנתונים על שאר הפרמטרים: עלות, זמן, מאמץ. להלן ההערכה המקובלת לגודל מחסן נתונים:

- ❖ **מחסן נתונים קטן:** היקף נתונים עד 3GB (גיגה-בתים), מספר השורות הכולל הוא כ- 10 מיליון, הטבלה הבודדת הגדולה ביותר היא בהיקף של כ- 5 מיליון שורות. מחסן כזה מיושם בדרך כלל על שרת PC חזק.
- ❖ **מחסן נתונים בינוני:** היקף נתונים בין 3 ל- 25 ג׳יגה בתים, מספר השורות הכולל הוא כ- 100 מיליון, הטבלה הבודדת הגדולה ביותר היא בהיקף של עד 50 מיליון שורות. מחסן כזה מיושם על שרת בודד חזק.
- ❖ **מחסן נתונים גדול:** היקף נתונים בין 25 ל- 200 ג׳יגה בתים, הטבלה הבודדת הגדולה ביותר יכולה להכיל מאות מיליוני שורות. מחסן כזה מיושם על שרת מרובה מעבדים.
- ❖ **מחסן נתונים ענק:** היקף נתונים מעל 200 ג׳יגה בתים. מחסן כזה מיושם על שרתים מרובי מעבדים, או על מחשב מרכזי.

מכיון שמחסן הנתונים צובר נתונים בהתמדה, קביעת היקף הנתונים רלוונטית לשלבים הראשוניים בלבד. מחסן נתונים שמוגדר כבינוני בעת הקמתו יכול להפוך במהירות לגדול ובהמשך - לענק. הלקח החשוב הוא שבכל מקרה יש לבנות את מחסן הנתונים מתוך צפייה ויכולת גידול (Scaleability) לאורך זמן, ללא קשר להיקפו הראשוני.

בחינת המוכנות הארגונית (Organizational Readiness)

נוסף לבחינת הצורך העסקי וההצדקה לביצוע פרויקט מחסן נתונים, חשוב לבחון בשלב זה את רמת המוכנות של הארגון ליישום תפיסת מחסן הנתונים. המוכנות צריכה להיבחן במספר מישורים: רמת המשתמשים הצפויים ויכולתם להתמודד עם כלי ניתוח המידע ורמת המוכנות של יחידת המחשב להתמודד עם טכנולוגיה חדשה.

הארגון צריך לסקור בתחילת הדרך את הפערים בין המצב הקיים לבין המצב העתידי שבו יפעל מחסן נתונים. פערים אלה עשויים להיות בידע ויכולת המשתמשים להפעיל כלי ניתוח מידע ופערים בטכנולוגיית המידע של הארגון. לדוגמה ניקח ארגון שאין לו כל ניסיון עם טכנולוגיית שרת/לקוח וכל מערכות המידע שלו מבוססות על מחשבים מרכזיים שמחוברים אליהם מסופים פשוטים. בארגון כזה יהיה צורך לקלוט מיומנויות חדשות, לפרוס רשתות תקשורת מהירות, להחליף את המסופים במחשבים אישיים, להכשיר את המשתמשים לעבודה בממשקים גרפיים וכדומה. ככל שהדברים יהיו ידועים בתחילת ההקמה, כך יוכל הארגון לטפל בהם ביתר הצלחה ולבצע את ההשקעות להקטנת הפערים. הבנת רמת המחויבות הנדרשת תתרום גם כן להצלחה בעתיד.

הצגת הצורך העסקי והמוכנות הארגונית להנהלה

המסמך המסכם של שלב בחינת הצורך העסקי והמוכנות הארגונית צריך להיות מוצג להנהלה הבכירה של הארגון. כך תוכל ההנהלה לקבל את ההחלטות על כניסה לפרויקט ולספק את המחויבות הנדרשת להצלחתו.

שלב 2: גיבוש ארכיטקטורת העל

מהות השלב

שלב זה עוסק בגיבוש ארכיטקטורת העל של מחסן הנתונים. מקובל לקרוא למסמך ארכיטקטורת העל המתקבל משלב זה גם בשם **תוכנית האב של מחסן הנתונים**. תוכנית זו מהווה בסיס ומסגרת לכל הפעולות העתידיות בתהליך ההקמה.

מסמך תוכנית האב מגדיר את הנושאים העיקריים שינוהלו במחסן הנתונים ואת הנתונים העיקריים הנדרשים עבור כל אחד מהם. בנוסף, הוא מפרט את המערכות התפעוליות שתשמשנה מקור עיקרי לנתונים; הוא מגדיר את הארכיטקטורה הטכנולוגית הכוללת וקובע מספר סטנדרטים ועקרונות מנחים; הוא מזהה את העלויות הצפויות ומגדיר לוח זמנים ליישום מחסן הנתונים. מטרת תוכנית האב להציג את האסטרטגיה הכוללת של הארגון בתהליך יישום מחסן הנתונים, ולאפשר להנהלה הבכירה לקבל את התמונה הכוללת לצורך החלטה על צורת מימוש הפרויקט.

היתרונות העיקריים של גיבוש תוכנית אב כשלב ראשון בפרויקט מחסן נתונים:

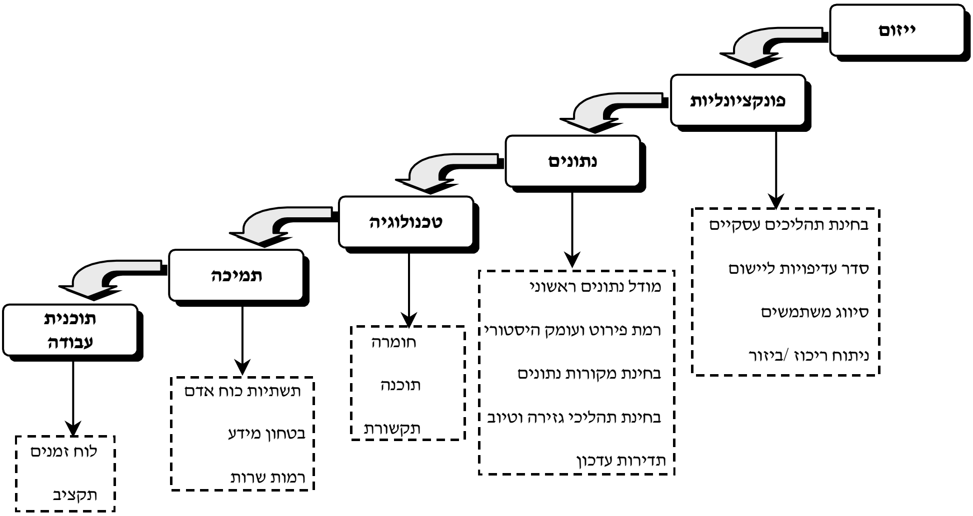
❖ **שיקוף צרכי הארגון:** תוכנית האב הינה מנגנון, שמטרתו להבטיח שמחסן הנתונים ישקף באופן אמיתי את הצרכים העסקיים של הארגון. אלה צריכים לבוא לידי ביטוי מיעדיו האסטרטגיים של הארגון ומתוך מסמך בחינת הצורך העסקי והמוכנות הארגונית, ולא רק כתוצאה מרצון כזה או אחר של אוכלוסיית משתמשים ושל יחידת המחשב.

❖ **אחידות:** תוכנית האב יכולה להבטיח אחידות בפלטפורמות, בכלים ובשיטות העבודה. הסכנה הגדולה ביותר היא עבור הארגונים המתחילים ביישום מרכולי נתונים, מבלי לבחון את התמונה הכוללת ואת שילוב כל מרכיבי המידע של הארגון. הארגון יכול למצוא את עצמו מהר מאוד במצב של מגדל בבל, שבו מחסני נתונים שונים מבוססים על כלים שונים ותקנים שונים, אשר בסופו של דבר אינם מאפשרים את שילוב הנתונים וכך, קבלת תמונה ארגונית כוללת הופכת לבלתי אפשרית.

❖ **מסלול התקדמות יישום ברור:** תוכנית האב מגדירה מסלול ברור ושיטתי להקמת מערכת תומכת החלטות ארגוניות תוך הגדרת העדיפויות, המשאבים והתוצרים בשלבים השונים. מבט העל מאפשר להגדיר את הסדר הנכון לשחרור מקטעים לשימוש, מבחינת התועלת לארגון ויכולת הארגון להקצות עבורם את המשאבים הדרושים.

תהליך הקמת תוכנית האב של מחסן הנתונים אינו תהליך חד-פעמי אלא מאמץ מתמשך. מחסן הנתונים משנה את פניו ללא הפסק לאורך מחזור חייו, ולכן רצוי לעדכן מעת לעת את תוכנית האב כדי שתשקף באופן אמיתי את הארכיטקטורה הכוללת של מחסן הנתונים בכל נקודת זמן.

התרשים הבא מציג את הנושאים העיקריים בהם יש לטפל במסגרת גיבוש אסטרטגיית העל. כל אחד מהנושאים מפורט בהמשך.



תרשים 12.4: הנושאים העיקריים המטופלים במסגרת שלב תוכנית האב.

ייזום פרויקט ארכיטקטורת העל (Project Charter and Work Plan)

חשוב להתייחס אל תהליך גיבוש ארכיטקטורת העל של מחסן הנתונים כאל פרויקט בפני עצמו. משך הקמת תוכנית האב קצר, בדרך כלל בין חודש לשלושה חודשים בהתאם לגודל הארגון ומורכבות מחסן הנתונים שיוקם, אולם הוא דורש משאבים וניהול כמו כל פעילות פרויקטלית אחרת. לשלב ראשוני זה יש חשיבות מרובה לתאום ציפיות בין הפרויקט לבין הנהלת הארגון: מה יהיו תוצרי תוכנית האב, מהו לוח הזמנים, מהם המשאבים הנדרשים וכדומה. **מסמך ייזום פרויקט** (Project Charter) מתאר בקצרה את שלבי הפרויקט ואת המשתתפים מהאגפים ומחלקות הארגון. הוא גם נותן תיאור תמציתי של פרקי המסמך המסכם ואת לוח הזמנים והמשאבים הנדרשים להשלמתם.

לאחר פרסום מכתב הייזום הממנה את מנהל הפרויקט ואת צוות הפרויקט, מתחיל תהליך התארגנות קצר להכנת תוכנית האב. עבור רוב המשתתפים בצוות גיבוש תוכנית האב זוהי חוויה מקצועית חדשה בדרך כלל. לכן יש להתחיל בהדרכה, שבה מוסברים לכל חברי הצוות העקרונות ומושגי היסוד של תפיסת מחסני נתונים, המתודולוגיה הכוללת להקמתו והנושאים המטופלים במסגרת שלב ראשון זה. בתום שלב ההדרכה יש להכין את תוכנית העבודה הצפויה ולפרט את המטלות ואת לוחות הזמנים של כל חבר צוות.

סקירת הנושאים המטופלים בתוכנית האב

הנושאים השונים המטופלים במסגרת שלב ארכיטקטורת העל מסווגים לפי ארבע קטגוריות: פונקציונליות, נתונים, טכנולוגיה ותמיכה. כל הנושאים מטופלים ברמת על בלבד ללא ירידה לפרטים ולרמת דיוק. הירידה לפרטים תבוא בשלבים הבאים במחזור החיים של מחסן הנתונים. נסקור כאן את אוסף הנושאים שיש לטפל בהם במסגרת תוכנית האב.

פונקציונליות

בפרק זה של תוכנית האב סוקרים את התהליכים העסקיים שבהם מחסן הנתונים צריך לתמוך וקובעים עבורם את סדרי העדיפויות ליישום, את סוגי המשתמשים העיקריים הצפויים במחסן הנתונים. בשלב זה גם עוסקים בניתוח אחת ההחלטות העיקריות שיש לקבל: הארכיטקטורה העקרונית שבה הארגון בוחר לממש את מחסן הנתונים. הארגון יכול לבחור מחסן נתונים ארגוני, אוסף של מרכולי נתונים בלתי תלויים, או מחסן נתונים רב-שכבתי המשלב את המחסן הארגוני עם אוסף מרכולי נתונים המוזנים על ידי המחסן הארגוני.

א. סקירת התהליכים העסקיים העיקריים

במסמך בחינת הצורך העסקי והמוכנות הארגונית מופיעים התהליכים העסקיים העיקריים שמחסן הנתונים צריך לתמוך בהם. תהליכים עסקיים אלה הם הבסיס הפונקציונלי, או במילים אחרות - **הגדרת הדרישות של מחסן הנתונים**. במידת הצורך יש לבצע במסגרת מסמך תוכנית האב בחינה מעמיקה של התהליכים העסקיים שזוהו במסמך בחינת הצורך העסקי. אם לא נדרשת העמקה נוספת, יועתקו התהליכים העסקיים וסדרי העדיפויות ליישום אל מסמך תוכנית האב, כדי להבטיח את שלמות המסמך ללא צורך להפנות את הקוראים למסמכים אחרים.

ב. סקירת סדרי עדיפויות ליישום

בדומה לתהליכים העסקיים שכבר הוצגו במסמך בחינת הצורך העסקי והמוכנות הארגונית, גם סדרי העדיפויות ליישום מוצגים כאן מחדש, לצורך השלמות של מסמך תוכנית האב. אם עבר זמן רב יחסית מאז שמסמך בחינת הצורך העסקי והמוכנות הארגונית הוצג להנהלה, מומלץ לבחון מחדש את סדרי העדיפויות ליישום, כדי לוודא שאכן העדיפויות לא השתנו.

ג. סיווג סוגי המשתמשים

אחד המאפיינים של מחסן הנתונים הוא הגיוון הרב בסוגי המשתמשים שלו (Data Warehouse User Types). להבדיל ממערכות תפעוליות, בהם המשתמש הינו בדרך כלל בדרג התפעולי ובדרג הביניים, במחסן הנתונים משתמשים כמעט כל מקבלי ההחלטות בארגון. לכל סוג משתמש יש בדרך כלל צורך שונה להצגת המידע ולניתוחו, ולכן צריך להתאים את כלי האחזור וניתוח המידע הרלוונטיים לסוגי המשתמשים השונים. משתמשים רבים יכולים להשתמש ביותר מאשר בכלי אחזור וניתוח מידע אחד.

הסיווג המקובל של משתמשי מחסן הנתונים הוא לפי הקטגוריות הבאות:

❖ **דרג זוטר.** מחסן הנתונים פונה בדרך כלל אל המנהלים בדרג הביניים ובדרג הבכיר, אך הוא יכול לשמש גם את אוכלוסיית המנהלים בדרג הזוטר. אוכלוסיית משתמשים זו מאופיינת בכך שהאינטראקציה שלה עם מחסן הנתונים היא פשוטה בדרך כלל ומבוססת על פי רוב על שאילתות קבועות מראש. בחלק מהמקרים משתמשים אלה אינם פונים ישירות למחסן הנתונים, אלא אל **מאגר הנתונים התפעולי** (Operational Data Store - ODS).

❖ **מנהלי ביניים (Middle Management).** אוכלוסיית משתמשים זו מאופיינת בכך שהיא מפעילה שאילתות מזדמנות, אולם רמת התחכום שלהן סבירה. הם זקוקים לכלים גרפיים נוחים להצגת נתונים, שיאפשרו להתמצא במחסן, לבנות שאילתות באופן עצמאי ולהפעילן בעת הצורך.

❖ **מנתחי מידע (Analyst).** אוכלוסיית משתמשים זו עוסקת בדרך כלל בניתוח מידע עבור מקבלי החלטות. מנתחי המידע מבצעים ניתוחים מורכבים מאוד, בעלי אופי מזדמן וברמת דחיפות גבוהה יחסית. לאוכלוסיית משתמשים זו דרושים כלים מתוחכמים בעלי יכולות סטטיסטיות מתקדמות, כלים לבניית תרחישים (What If Analysis) וכלים לכריית נתונים (Data Mining).

❖ **מנהלים בכירים (Executives).** אוכלוסייה זו מאופיינת בדרך כלל ברתיעה מסוימת ממחשבים. בעלי תפקידים אלה מוגבלים מאוד בזמן שהם יכולים להקדיש לניתוח מידע. על כן יש להכין עבורם שאילתות מוכנות מראש, עם גרפיקה עסקית טובה ויכולות נבירה (Drill Down) בתוך הנתונים מרמת סיכום גבוהה מאוד ועד לרמות נמוכות יותר.

הטבלה הבאה מפרטת את כלי השימוש (Front End) המתאימים לכל סוג משתמשים:

קטגוריה	דרישות מידע	סוג כלי Front End
דרג זוטר	שאילתות ודוחות פשוטים יחסית, דרישות פרמטריות שניתנות לחיזוי מראש.	אוסף שאילתות מוכנות מראש עם כלי פיתוח גרפיים.
דרג ביניים	שאילתות מזדמנות, דוחות, ניווט במחסן הנתונים, גרפיקה עסקית.	כלי שאילתות גרפיים למשתמשי קצה, מחוללי דוחות.
מנתחי מידע	ניתוחים רב-מימדיים, ניתוחים סטטיסטיים מורכבים, ניתוח מגמות.	כלי ניתוח רב-מימדיים, כלים סטטיסטיים, גיליונות אלקטרוניים, כלי גרפיקה, כלי כריית נתונים.
מנהלים בכירים	מידע מוכן מראש על נושאים מסוימים, יכולת לרדת ברמת הפירוט ולנבור בנתונים.	כלי EIS.

על סמך ניתוח וסיווג משתמשי מחסן הנתונים במסגרת תוכנית האב, ניתן יהיה בהמשך לקבוע איזה סוגים של כלי משתמש יידרשו ולאילו מטרות.

ד. ניתוח ריכוז/ביזור

אחת ההחלטות החשובות בגיבוש תוכנית האב של מחסן הנתונים קשורה בנושא ריכוז/ביזור (Centralized / Decentralized Warehouse). תפיסת ריכוז/ביזור קשורה לקביעת הארכיטקטורה של מחסן הנתונים. הארגון צריך להחליט מהי הארכיטקטורה המועדפת עליו, מבין שלוש החלופות הבאות:

- ❖ מחסן נתונים ארגוני המשרת את כל המשתמשים ומכיל את כל סוגי הנתונים ונושאי המידע.
- ❖ אוסף של מרכולי נתונים מבוזרים המשרתים מקטעי משתמשים שונים (בחלוקה גיאוגרפית, חלוקה פונקציונלית, נושאית או אחרת). מרכולים אלה מוקמים באופן עצמאי, ללא תאום ובאופן בלתי תלוי אחד בשני.
- ❖ מחסן נתונים רב-שכבתי המורכב ממחסן נתונים ארגוני, המזין אוסף של מרכולי נתונים מבוזרים. בארכיטקטורה זו מרכולי הנתונים נבנים במקביל להקמת מחסן הנתונים הארגוני ובהתבסס על ארכיטקטורה כוללת המוגדרת בשלב ההקמה.

הארכיטקטורה המומלצת ביותר כיום היא הארכיטקטורה הרב-שכבתית המשלבת את הדרישות ברמה הארגונית יחד עם מתן מענה ליחידות הארגוניות הבודדות.

בפרק 11, העוסק במרכולי נתונים, הוצגו מספר שיטות הקמה. ההמלצה היא לבחור במודל ההקמה המקבילי.

הנתונים

פרק זה בתוכנית האב עוסק בנתונים, תוך הצגת מודל נתונים ראשוני, סקירה של מקורות הנתונים, תדירות הבאת הנתונים מהמערכות התפעוליות אל מחסן הנתונים ועוד.

א. בניית מודל הנתונים הראשוני (Initial Data Modeling)

לאחר איתור והבנת התהליכים העסקיים בונים מודל ראשוני של מחסן הנתונים הנדרש. לעיצוב משתמשים בסמלים הלקוחים מ**תרשימי יישויות-קשרים** (Entity Relationship Diagram - ERD), כמו יישויות, תכונות, מפתחות, קשרים. במודל זה מציגים אותם במבנה ייחודי למחסני נתונים: סכמות כוכב (Star Schema), סכמות פתיתי שלג (Snowflake Schema) ועוד. מודלים אלה פותחו במיוחד עבור מחסני נתונים ומטרתם לבנות מבנה נתונים פשוט מנקודת מבט המשתמש, אך עם זאת בעל ביצועים טובים. נדגיש שבשלב זה אין לבנות מודל מדויק, אלא רק מודל ראשוני מאוד המציג את יישויות הנתונים העיקריות במבט-על. מודל הנתונים הראשוני יעודכן עם הזמן, בעת ההתקדמות במחזור החיים אל השלב הבא: עיצוב ויישום של המקטעים השונים במחסן הנתונים.

ב. רמת הפירוט והעומק ההיסטורי (Granularity and Retention Policy)

אחת העוצמות הגדולות ביותר של מחסן הנתונים נובעת מיכולתו לנהל נתונים מפורטים לפרקי זמן ארוכים. יחד עם זאת, קיים מבחן של עלות-תועלת גם בנושא זה. שמירת נתונים מפורטים לאורך זמן הינו תהליך יקר הדורש משאבי מחשוב מרובים ויש לו

השפעות רבות על הביצועים של השאילתות. בשלב מוקדם דרושה החלטה על רמת הפירוט הנדרשת עבור סוגי הנתונים השונים.

לעומק ההיסטורי של הנתונים שינוהלו במחסן הנתונים יש השלכה על גודל בסיס הנתונים, על סוגי ניתוחי המידע שניתן יהיה לבצע ועל מדיניות הגיבוי והשחזור. ככל שהעומק ההיסטורי רב יותר, היקף הנתונים שינוהל גדול יותר שיאפשר ניתוחי מגמה רב-שנתיים. לעומת זאת, ניתן לקבוע עומק היסטורי קצר שיפגע במידה מסוימת באפשרות לבצע ניתוחי מידע רב-שנתיים. העומק ההיסטורי יכול להיות שונה עבור תקופות זמן שונות. כך למשל, ניתן לקבוע שנתוני השנה האחרונה ינוהלו באופן המפורט ביותר, ונתוני שנים קודמות ינוהלו בצורה סיכומית בלבד.

ניתן בשלב זה להחליט על רמה בסיסית של עומק היסטורי בעת הקמת מחסן הנתונים ולאחר מכן להרחיב את רמת הפירוט והעומק שלה. אחת המטרות העיקריות של המתודולוגיה היא להקטין סיכונים בכישלון הפרויקט ולאפשר הקדמת תוצרי ביניים. על כן, אפשר להחליט שבשלב ראשון העומק ההיסטורי יהיה נמוך (למשל שני רבעונים בלבד) ורק לאחר תקופה מסוימת של בחינת מחסן הנתונים העומק ההיסטורי יורחב לשנה ולאחר מכן לשנתיים. להחלטות אלו יש משמעות מבחינת התועלות שניתן להפיק ממחסן הנתונים בשלבי הפרויקט הראשוניים. לעיתים כדאי להקריב חלק מהתועלות על חשבון היכולת לטפל במחסן נתונים בעל גודל סביר, לפני שעוברים לנהל בסיסי נתונים ענקיים - VLDB (Very Large Data Base).

ג. בחינת מקורות הנתונים (Data Sources Analysis)

בשלב זה בוחנים ומאתרים את מקורות הנתונים הדרושים לתמיכה בנושאי המידע ובתהליכים העסקיים שהוגדרו בשלב הקודם. ליישום מחסן הנתונים דרושים מקורות נתונים שונים שחלקם מצויים במערכות התפעוליות של הארגון וחלקם מצויים במערכות חיצוניות לארגון (כמו סקרי שוק, דוחות של הלשכה המרכזית לסטטיסטיקה וכדומה).

הפלט הסופי של שלב זה הינו מטריצה של נושאים/מקורות, המגדירה לכל נושא את מקורות הנתונים הרלוונטיים עבורו. לכל מקור שיאותר יש להגדיר מספר מאפיינים, וביניהם:

- איזו מערכת מידע תפעולית מנהלת את מקור הנתונים.
- מה צורת ותדירות העדכון.
- סביבת החומרה בה מנוהל מקור הנתונים.
- סביבת התוכנה (מערכת הפעלה, מערכת לניהול בסיסי נתונים, מערכת תקשורת וכדומה).
- אילוצים מיוחדים.

ד. בחינת תהליכי שליפה וגזירה של הנתונים (Source to Target Mapping)

גזירת הנתונים הדרושים ממחסן הנתונים היא הפעולה הראשונה בדרכם של הנתונים מהמערכות התפעוליות אל מחסן הנתונים. בתהליך השליפה והגזירה של הנתונים דרוש מנגנון מיוחד שיבטיח שהם עקביים, עדכניים ומאורגנים במבנה מתאים. בתהליך זה מתבצעות פעולות נוספות, הכוללות את הכנת הנתונים לקראת העברתם למחסן הנתונים, וביניהם:

- ❖ בדיקה האם הנתונים אמורים להחליף נתונים אחרים במחסן או להתווסף אליהם.
- ❖ ניקוי וטיוב הנתונים.
- ❖ סינון נתונים.
- ❖ סיכומים וחישובים.
- ❖ פירוק הנירמול שלהם (Denormalization).
- ❖ הוספת מימד הזמן לנתונים.
- ❖ שילוב נתונים ממקורות שונים (דואר אלקטרוני, מאגרי מידע חיצוניים וכדומה).

לאחר הגדרת מקורות הנתונים ותהליכי ההתמרה (טרנספורמציה) שעליהם לעבור, ניתן לבחון בכלים קיימים המסוגלים לבצע את הפעולות שהוגדרו. בכל מקרה, נשתמש בכלל אצבע פשוט: ככל שתהליכי העיבוד וההתמרה מורכבים יותר, הסיכוי לבצע אותם באמצעות כלים אוטומטיים הולך ופוחת. לכך תידרשנה תוכניות מיוחדות שתיכתבנה בשפת תכנות סטנדרטית כלשהי. מקובל גם להניח שבשלב ראשון רצוי לפתח תוכניות מיוחדות ורק לאחר לימוד התנהגות הנתונים ומורכבות העיבודים לחפש כלים לאוטומציה.

ה. הגדרת תדירות העדכון והעברת הנתונים

באופן טבעי, הנטייה היא לבנות מחסן נתונים שרמת העדכניות שלו גבוהה. טעינת מחסן הנתונים מתבצעת בתהליכי גזירה חוזרים שלאחריהם טעינת הנתונים שנגזרו. ניתן להגיע לרמת עדכניות גבוהה, ואפילו יומית. אפשר לחזור על פעולה זו גם מספר פעמים ביום, אך בדרך כלל אין צורך לעשות זאת. מחסן הנתונים משמש לתמיכה בתהליכי קבלת החלטות, ואינו מיועד לשרת צרכים תפעוליים שוטפים, ולכן סביר להניח שרמת עדכניות יומית נראית סבירה.

עם זאת, חשוב לבחון סוגיה זו, כדי ללמוד האם באמת דרושה רמת עדכניות יומית. העדכון היומי של מחסן נתונים כרוך במאמץ תפעולי רב, ולכן צריך לבדוק האם הוא כדאי. לעיתים מתברר שבגלל אופי הנתונים, או בגלל אופי ניתוחי המידע, גם עדכניות שבועית, חודשית ואפילו רבעונית מספיקים בהחלט. לדוגמה, במערכת בנקאית מבצעים חישובי ריביות רק פעם בחודש, ולכן אין כל משמעות לביצוע ניתוחים במהלך החודש.

הגדרת נפחי הנתונים המועברים יכולה להכתיב גם כן את תדירות ההעברה. לדוגמה, בחברת טלפונים מצטברים מדי יום עשרות ומאות מיליוני רשומות של פרטי שיחה - CDR (Call Detail Record). העברת נתונים בכמויות אלו פעם בשבוע עלולה לגרור זמן עדכון לא סביר, ולכן צריך לשקול עדכון יומי של מחסן הנתונים.

טכנולוגיה

א. הגדרת מרכיבי הארכיטקטורה הטכנולוגית

בשלב זה יבוצע עיצוב העל של הארכיטקטורה הטכנולוגית (Technological Architecture Blueprint) הכוללת את הגדרת התשתיות השונות הנדרשות והאינטגרציה ביניהם. מטרת הארכיטקטורה הטכנולוגית להציג את התמונה הכוללת של כל המרכיבים הנדרשים ליישום מחסן הנתונים, הן בטווח הקצר והן בטווח הארוך. מטרת תהליך עיצוב זה במסגרת תוכנית האב איננה לקבל החלטות סופיות לגבי כל אחד מהמרכיבים הטכנולוגיים (כמו בסיס הנתונים, מערכת ההפעלה של השרת, מהם כלי גזירת הנתונים וכדומה), אלא בעיקר לבחון את הטכנולוגיות הקיימות כיום בארגון ואת מידת התאמתן לצרכים, ולקבל אומדן סביר לגבי היישימות והעלויות.

דוגמה לבחינת מרכיבי הארכיטקטורה היא ארגון שמבוסס רק על מחשבים מרכזיים ומבקש לראשונה ליישם פרויקט בקנה מידה משמעותי עם מערכת הפעלה מסוג Unix על שרתים מרובי-מעבדים (SMP או MPP). דוגמה אחרת היא ארגון המבקש ליישם מחסן נתונים מבוזר, שבו השרת העיקרי יבוסס על מערכת הפעלה Unix בעוד השרתים של מרכולי הנתונים יתבססו על Windows NT.

החלטות מפורטות לגבי כל אחד מהרכיבים יכולות להתקבל בשלב מאוחר יותר, לאחר אישור תוכנית האב. מטרת הצגת התמונה הארכיטקטונית של מחסן הנתונים בשלב ראשוני זה היא לאפשר לארגון לבחון את כל המשמעויות ואת צורת ההתארגנות שלו לקראת יישום הטכנולוגיות החדשות שעליהן יתבסס מחסן הנתונים.

לסיכום, שתי החלטות עקרוניות צריכות להתקבל בשלב זה: מחסן נתונים מרכזי, מרכול נתונים או שילוב ביניהם ורמת הפירוט והעומק ההיסטורי של הנתונים. על בסיס החלטות אלו ניתן לבחון קשת שלמה של נושאים טכנולוגיים, אשר ישפיעו על ארכיטקטורת מחסן הנתונים. נפרט את החשובים שבהם:

- ❖ **בסיס הנתונים (Data Base):** יש להגדיר על איזה בסיס נתונים יבוסס מחסן הנתונים. רוב מחסני הנתונים מיושמים כיום על מערכות RDBMS, אולם עדיין עלינו לבחון אם היישום יהיה על שרת מסוג Unix, על שרת NT או על שרת אחר (כמו DB2 על מחשב יבמ מרכזי, או DB2/400 על מחשב AS/400 וכו'). אחד הפרמטרים העיקריים שיש לשקול בהחלטה זו הוא היקף הנתונים שינוהלו: מספר עשרות GigaByte, מספר מאות GigaByte או TeraByte ומעלה. פרמטר נוסף מקובל הוא מידת הפתיחות של בסיס הנתונים לכלי שאילתות ומחוללי הדוחות הנפוצים בשוק.

- ❖ **שרת מחסן הנתונים (Data Warehouse Server):** יש לקבוע את ארכיטקטורת השרת וכושר הגידול הנדרש ממנו לטווח התכנון. השרת יכול להיות מבוסס על ריבוי מעבדים בארכיטקטורת SMP, MPP או NUMA. אחד הנושאים העיקריים שיש לתת עליו את הדעת בשלב זה הוא כושר הגידול הנדרש, או מידרגיות (Scaleability), כנגזרת מגודל בסיס הנתונים ומזמני התגובה הנדרשים. אם הארגון מתכוון לבנות מחסן נתונים מבוזר, צריך להגדיר את השרת העיקרי ואת שרתי הנתונים של מרכולי הנתונים המבוזרים.

- ❖ **שרת התמרת נתונים (Transformation Server):** בעת התכנון אנו בוחנים את היקף הנתונים הנגזרים מהמערכות התפעוליות, בודקים את הפלטפורמות השונות והרבות המשרתות את המערכות התפעוליות ורואים את מורכבות תהליכי הניקוי וההכנה של הנתונים לקראת טעינתם למחסן הנתונים. כדי לפתור בעיות אלו עלינו להשתמש לא אחת בשרת ייעודי שתפקידו לקבל נתונים במצב גולמי מהמערכות התפעוליות ולבצע עליהם את בדיקות התקינות, להשלים נתונים חסרים, לפענח קודים ועוד. ביצוע תהליכים אלה על שרת ייעודי מקטין את העומס על שרת הנתונים של מחסן הנתונים ולעיתים זוהי הדרך היחידה לעמוד בחלון הזמן הקיים לטעינת הנתונים. מקובל לקרוא לארכיטקטורה זו Hub Architecture, כי כל זרימת נתונים מהמערכות התפעוליות אל מחסן הנתונים עוברת דרך תחנה מרכזית השולטת על תהליכי הזרימה ומבצעת את טיוב הנתונים הנדרש.
- ❖ **תקשורת הנתונים:** יש להגדיר את תצורת רשת התקשורת שתתמוך בהעברת הנתונים מהמערכות התפעוליות אל מחסן הנתונים, מהמחסן הארגוני אל מרכולי נתונים מחלקתיים וכדומה. רוחב הפס של תשתית התקשורת הינו גורם חשוב מאוד באמידת היכולת לעמוד באילוצי חלון הזמן הפנוי לרשות הארגון להעברת וטעינת הנתונים.
- ❖ **כלי ביניים (Middleware Software):** לעיתים צריך להשתמש בכלי תוכנה המשמשים כתווך בין פלטפורמות מחשוב שונות. לכלים אלה מיגוון תפקידים, כמו המרת קודים בין EBCDIC ל-ASCII, שיכפול נתונים בין בסיסי נתונים שונים (Replication Tools), תוכנות גישור לגישה לבסיסי נתונים שאינם בסיסי נתונים טבלאיים וכדומה.
- ❖ **כלי קצה (End User Tools):** על פי סוגי הנתונים שינוהלו במחסן הנתונים וסוגי האוכלוסיות של המשתמשים ניתן בשלב זה להגדיר ברמה העקרונית בלבד את סוגי כלי הקצה שיידרשו. אלה יכולים להיות כלי שאילתות ומחוללי דוחות פשוטים, כלים לניתוח מידע רב-מימדי, כלים סטטיסטיים, כלי כריית נתונים וכדומה. גם כאן ניתן להחליט על יישום מדורג, מהפשוט אל המורכב. תחילה מפעילים כלי קצה פשוטים ולאחר מכן עוברים לכלים מתוחכמים יותר. כאשר יודעים בשלב תוכנית האב מהם כלי הקצה שיידרשו, הארגון יוכל להיערך בהתאם.
- ❖ **קטלוג הנתונים (Meta Data/Repository)**: קטלוג זה כולל את הנתונים המנוהלים במחסן הנתונים וייעודו לאפשר למשתמשים להתמצא: מה נמצא, היכן, למתי זה מעודכן, מהם מקורות הנתונים, מה קשור עם מה וכו׳. בשלב זה בודקים את הכלים השונים היכולים לשמש כבסיס לניהול קטלוג המידע. צריך לשקול בשלב זה אם ליישם את הקטלוג מייד, או שניתן לדחות החלטה זו לשלב מאוחר יותר.

תמיכה

בפרק זה של תוכנית האב סוקרים מספר נושאים הקשורים לצורת התמיכה במחסן הנתונים: איזה כוח אדם נדרש לממש את הפרויקט, מהן סוגיות אבטחת המידע של מחסן הנתונים שיש לדון בהן ומה רמת השירות הדרושה על ידי המשתמשים השונים.

א. בחינת תשתיות כוח אדם הדרושות ליישום מחסן נתונים

ארגונים נוטים לעיתים להזניח את המשמעויות של מאפייני כוח האדם הדרושים ליישום מחסן הנתונים. ההנחה, המוטעית בדרך כלל, שממילא יש לנו מנהלי בסיסי נתונים (DBA), מפעילי מחשב ותוכניתנים - ולכן ניתן להצליח ביישום מחסן הנתונים ללא תוספת משאבי אנוש. לרוב המציאות טופחת על פני מקבלי החלטות אלה בשלבים מאוחרים יותר, ואז יש צורך להסביר כיצד זה נדרש "פתאום" גידול בכוח האדם. כדי למנוע הפתעות אלו, כדאי כבר בשלב תוכנית האב להתייחס לנושא זה ולתת אומדן ראשוני של כוח האדם הנדרש: סוג (כישורים) ומספר עובדים.

בשלב זה ניתן להגדיר את היקף כוח האדם על סוגיו השונים הדרושים ליישום מחסן נתונים בארגון:

- ❖ מספר מנהלי בסיסי הנתונים שיידרשו, הוא בדרך כלל פונקציה של גודל בסיס הנתונים ומורכבותו.
- ❖ מספר התוכניתנים המתחזקים את המערכות התפעוליות, שיידרשו לבניית תהליכי גזירת הנתונים מהמערכות התפעוליות.
- ❖ מספר התוכניתנים שיידרשו לבניית תהליכי הטיוב, ההתמרה וההכנה לקראת הטעינה.
- ❖ מספר התוכניתנים שיידרשו כדי לבנות יישומים ייעודיים להפעלת מחסן הנתונים.
- ❖ מספר מנתחי מערכות שיידרש כדי לתמוך באוכלוסיות המשתמשים השונות.

התייחסות לנושאי כוח אדם בשלב גיבוש תוכנית האב תאפשר לארגון להיערך בזמן, וגם להטמיע את מלוא המשמעות ועלות תפעול שדרושה בסביבת מחסן הנתונים.

ב. אבטחת מידע

מחסן הנתונים מכיל מיגוון סוגי מידע חיוני ורגיש מבחינת הארגון. בדומה לסביבות תפעוליות, גם במחסן הנתונים יש להגדיר איזה משתמשים יכולים לראות איזה נתונים. בשלב של האסטרטגיה יש לקבוע רק את עקרונות אבטחת המידע שהארגון מבקש לאמץ, את רמות המידור שיש ליישם וכדומה.

ג. רמות שירות

מחסן הנתונים יהיה חלק חשוב מאוד מניהול המידע בארגון וציפיות השירות ממנו תהיינה דומות לאלו שמצפים ממערכות תפעוליות. לפיכך יש להגדיר גם עבור מחסן הנתונים **הסכמי רמות השירות** (Service Level Agreements - SLA). יש להגדיר אם המחסן יפעל כל היממה או רק בשעות העבודה הרגילות ואיזה זמני תגובה יובטחו למשתמשים השונים. חלק מהמשתמשים זקוקים לזמני תגובה קצרים מאוד לחלק מהשאילתות והניתוחים ומוכנים להסתפק בזמני תגובה ארוכים יותר לחלק אחר של השאילתות.

תוכנית עבודה ליישום

פרק זה בתוכנית האב מציג את תוכנית היישום של פרויקט מחסן הנתונים בהתבסס על כל ההחלטות שהתקבלו ותוך הצגת אומדנים ללוחות זמנים, לעלויות ולמשאבי כוח אדם.

א. לוחות זמנים (Time Tables)

בשלב זה יוכנו לוחות הזמנים למימוש הפרויקט, הכוללים את פירוט שלבי הביניים ואת מהלך היישום של כל שלב כזה. המתודולוגיה מבוססת על שחרור חלקי של מקטעים שונים של מחסן הנתונים ולכן לוח הזמנים יציג את עיתוי היישום של המקטעים השונים.

ב. אומדן תקציב פרויקט (Project Budget Estimation)

בשלב זה ניתן לבנות את אומדן התקציב הכולל של הפרויקט. כמו בכל פרויקט, יש להבחין בין הוצאות חד-פעמיות (One Time Expenditures) לבין הוצאות שוטפות (Recurring Expenditures). נגדיר תחילה את הנושאים השונים שיש לתקצב לנושאים חד-פעמיים ולאחר מכן את סוגי ההוצאות השוטפות המקובלות בפרויקטי מחסן נתונים.

❖ **הוצאות חד-פעמיות:**

- **חומרה**: מעבדים, דיסקים, תקשורת, מחשבים אישיים, אמצעי גיבוי מיוחדים.
- **תוכנה**: מערכת לניהול בסיס הנתונים, כלים למשתמשי הקצה, כלים לניהול מערכת, כלים לגזירה והמרה של הנתונים, כלים לניהול קטלוג הנתונים.
- **כוח אדם**: המשאבים הנדרשים לתקופת הפרויקט, עלויות הדרכה לשימוש במוצרים החדשים.

❖ **הוצאות שוטפות:**

- עלויות תחזוקה של כל המוצרים שנרכשו.
- עלויות גיבוי הנתונים.
- עלויות הדרכה שוטפות.
- עלויות כוח אדם לתפעול שוטף של מחסן הנתונים, ובכלל זה מנתחי מערכות, מפתחים, אנשי תפעול, מנהלי בסיס נתונים, אנשי תקשורת.

עלויות משאבי המחשב של הסביבה התפעולית המוקדשים לתהליכי הגזירה וההתמרה.

השאלה המעניינת ביותר היא: כמה זה עולה. לצערי, התשובה על שאלה זו מורכבת וכמו תמיד - תלויה בלא מעט גורמים ייחודיים לכל ארגון וביניהם:

❖ גודל הארגון.

❖ הארכיטקטורה של מחסן הנתונים - מרכזי או מבוזר.

❖ רמת הפירוט של הנתונים ונפחם.

❖ כמות ההיסטוריה שהארגון מבקש לנהל.

- ❖ כמות הסיכומים שיש לנהל.
- ❖ רמת התחכום של משתמשי הקצה.
- ❖ מספר המרכולים שהארגון מבקש לבנות.
- ❖ רמת האוטומציה שהארגון רוצה להכניס בתהליכי הגזירה וההמרה.

מכל האמור לעיל עולה תמונה ברורה שאין מושג "ארגון טיפוסי" או "פרויקט מחסן נתונים טיפוסי". השונות בין הארגונים והפרויקטים יכולה להיות גדולה מאוד. המידע היחיד בתחום זה מתייחס לחלוקת עוגת התקציב בין סעיפי ההוצאה השונים. מתוך מחקרים שונים שנעשו על ידי חברות כגון Meta Group, עולה התמונה הבאה:

- ❖ אם מתייחסים לעלויות הפיתוח בלבד, החלוקה היא בערך שליש לכל אחד מהסעיפים חומרה, תוכנה ועבודה.
- ❖ אם מתייחסים למחזור החיים המלא של מחסן הנתונים, נקבל: חומרה - 15%, תוכנה - 13%, עבודה - 31% ותחזוקה - 41%.

שלב 3: עיצוב ויישום מקטע (Data Warehouse Increment Implementation)

מהות השלב

בגמר הכנת תוכנית האב של מחסן הנתונים ואישורה על ידי ההנהלה, מתחיל למעשה שלב העיצוב והיישום של מחסן הנתונים. ניתן לחלק שלב זה לשני שלבי משנה עיקריים:

- ❖ **בניית המקטע הראשון** של מחסן הנתונים, או כמו שהוא נקרא לעיתים **מקטע הבסיס**. במובן מסוים ניתן לראות את מקטע הבסיס כפרויקט ניסוי (Pilot) למימוש תפיסת מחסן הנתונים.
- ❖ בניית **המקטעים הבאים**, מקטע אחר מקטע ולפעמים מספר מקטעים במקביל.

בניית מקטע הבסיס (Baseline)

מקטע הבסיס אינו שונה ממקטעים אחרים, למעט העובדה שהוא הצעד הראשון של הארגון למימוש תפיסת מחסן הנתונים. במסגרת מקטע זה מתחיל הארגון להתנסות לראשונה בחוויה של בניית מחסן הנתונים ולכן חשיבותו נובעת מהלקחים שניתן להפיק ומהתשתיות שנבנות וישרתו את המקטעים הבאים. בהמשך נפרט מספר נושאים המדגישים את החשיבות וההתייחסות המיוחדת שיש לתת למקטע הבסיס.

- ❖ בשלב יישום מקטע הבסיס הארגון צריך להתקין את סביבת המחשוב החדשה. עבור חלק מהארגונים המשתמשים בפלטפורמות מחשוב אחרות (כגון מחשבים מרכזיים, מחשבי AS/400, מחשבי Alpha/VMS וכדומה), זו יכולה להיות התנסות ראשונה בהתקנה ובהפעלה של פלטפורמות מחשוב המבוססות על שרתי Unix או NT (עבור

מחסני הנתונים הקטנים יותר). ייתכן שזו תהיה התנסות ראשונה בהתקנת והפעלה של מערכת RDBMS, התנסות ראשונה בעיצוב בסיסי נתונים ייעודיים למחסני נתונים, התנסות ראשונה בכלי ניתוח מידע רב-מימדיים וכדומה. התנסות חדשה זו מחייבת גם תהליכי הכשרה של כוח אדם.

❖ בשלב יישום מקטע הבסיס נקבעים תקנים שונים, נקבעים נהלי עבודה חדשים, נקבעים כלי תשתית שישמשו את שאר המקטעים, ועוד. בגלל הראשוניות, טבעי הוא שנעשות שגיאות שנובעות בחלקן מחוסר מידע על דברים צפויים בהמשך הפיתוח, ויהיה צורך לבצע תיקונים שונים. היתרון נובע מהעובדה שאת הלקחים והניסיון שנצבר במקטע הראשון ניתן ליישם ביישום המקטעים הבאים. אין זה בא לומר שלאחר יישום מקטע הבסיס אין לומדים יותר וכל שנשאר זה רק ליישם. חלק מהטעויות מתגלות בשלבים מאוחרים יותר, וככל שמתרחקים מנקודת ההתחלה תהליכי התיקון עלולים להיות מורכבים יותר.

❖ לעיתים הארגון עדיין אינו משוכנע לחלוטין בהצדקה הכלכלית להפעלת מחסן הנתונים ואינו יכול להעריך את התועלת הצפויה. על כן, מקטע הבסיס משמש להרצת ניסוי של תפיסת מחסן הנתונים. הארגון מפעיל מקטע זה, בוחן את ההשקעות, את הטכנולוגיות שהוא מפעיל ואת התועלות השונות וצריך להשתכנע שהדרך שהוא הולך בה אמנם מסוגלת לתת מענה טוב לדרישות. רק לאחר שהארגון משתכנע ביישימות הפרויקט מתחיל תהליך בניית מחסן הנתונים.

עיצוב ובניית מקטע

בשלב זה מתבצע תהליך העיצוב המפורט של כל אחד מהשלבים שפורטו לכל אחת מקטגוריות הנושאים שמרכיבים את מחסן הנתונים. רוב הנושאים שמופיעים בשלב זה נדונו בהרחבה במסגרת הספר, ולכן לא נחזור על התיאור המפורט שלהם, אלא רק נציין את עיקרי המטלות שיש לבצע.

פונקציונליות

בשלב הזה מתבצע התכנון המפורט של סוגי ניתוחי המידע שיבוצעו מול מחסן הנתונים. ייבדק מי המשתמשים ואיזה כלים יועמדו לרשותם. במידת הצורך, יפותחו יישומים ייעודיים או שאילתות מוכנות מראש עבור אוכלוסיות משתמשים מסוימות.

נתונים

בשלב זה מתבצע העיצוב המפורט של הנתונים הכולל: מודל הנתונים המדויק, מיפוי מפורט של נתוני המקור, הגדרת כללי ההמרה של הנתונים, עיצוב מפורט של תוכניות הגזירה וההמרה, הגדרת שיטת ניהול המפתחות המלאכותיים ועיצוב המנגנונים לניהול מפתחות אלה, עיצוב תהליכי הטעינה של הנתונים אל מחסן הנתונים, הגדרת שיטת הטיפול בשגויי טעינה, תהליכי בניית האינדקסים של מחסן הנתונים, עיצוב הסיכומים שינוהלו במחסן הנתונים וצורת בניית הסיכומים.

בגמר תהליכי העיצוב מתחיל תהליך הבנייה והפיתוח של מרכיבי התוכנה השונים ומתבצעים מבחני קבלה של מרכיבי תוכנה אלה.

ארכיטקטורה טכנולוגית

בשלב זה מתבצעת הגדרה מדויקת של סביבות החומרה, התוכנה והתקשורת של הפרויקט. גם מבוצע רכש במידת הצורך ומבוצעת ההתקנה. לאחריה נעשית הדרכה לשימוש בכלים השונים ובסופם - מבחני אינטגרציה של סביבות הפיתוח והייצור החדשות כדי לוודא שהן תקינות ומוכנות להפעלה.

תמיכה

בשלב זה מגדירים במפורט את סוגי משאבי כוח האדם הנדרשים לפרויקט ואת מספר האנשים הדרושים, מגייסים עובדים כנדרש (מחוץ לארגון או מתוכו), נמשכת הדרכה לשימוש בכלי העבודה מול מחסן הנתונים, מוגדרים שיטות התפעול של סביבת מחסן הנתונים.

שלב 4: תפעול, תמיכה ושיפור

מהות השלב

בשלב זה מתחילים בתפעול שוטף של מקטע מסוים, על כל המשתמע מכך : תפעול תהליכי הגזירה והטעינה, תפעול כלי המשתמש וכדומה. עכשיו מתחילים בניטור שוטף של ביצועי השאילתות השונות, מתבצעים תהליכי כוונון כפי הצורך, עורכים שינוי במבנה הנתונים כתוצאה מדרישות שוטפות, מלווים את הפרויקט וממשיכים בהדרכה שוטפת למשתמשים, ועוד.

מספר עצות מעשיות בבניית מחסן הנתונים

קל מאוד לאבד את המיקוד בעת בניית מחסן הנתונים, אם יוצאים מתוך ההנחה, המוטעית כמובן, שמחסן הנתונים צריך להכיל כל פיסת נתון שהארגון מנהל. האמת היא כמובן שונה : **מחסן הנתונים צריך לנהל את אותם נתונים המשמשים את תהליכי קבלת ההחלטות**. הסיכויים להצליח ביישום יהיו הרבה יותר גבוהים, אם מחסן הנתונים יכיל וישלב רק את אותם נתונים שהם בעלי הערך הגבוה ביותר בתהליכי קבלת ההחלטות. הדרך הנכונה היא למצוא את המספר הקטן ביותר של מקורות נתונים שיתנו את מירב התועלת - קל להגיד אבל קשה לממש. ננסה להציג בהמשך אוסף של עצות מעשיות, שיאפשרו לך לבנות את מחסן הנתונים באופן ממוקד ובסופו של דבר יגדילו את סיכויי הצלחתו.

השיקולים להקמת מחסן הנתונים חייבים להיגזר מאסטרטגיית הארגון

מחסן הנתונים אינו פרויקט טכנולוגי שמונחה על ידי יחידת המחשב של הארגון. לפנינו **פרויקט ארגוני**, ובתור שכזה הוא חייב להתבסס על היעדים האסטרטגיים של הארגון ולהיות מסונכרן אתם. רק מחסן נתונים המתואם עם היעדים האסטרטגיים ותומך בהם באופן ישיר הוא מחסן שיצליח ויצדיק את ההשקעה בו. אל תחסוך במאמץ הנדרש בשלב בחינת הצורך העסקי ובשלב גיבוש תוכנית האב, כדי לזהות את היעדים האסטרטגיים של הארגון ולהבין כיצד מחסן הנתונים יתמוך בהם.

האחריות לביצוע הפרויקט

אחת השאלות שעולות תדיר בפרויקטים בכלל ובפרויקטי מחסן נתונים בפרט היא, מי צריך לעמוד בראש הפרויקט. התשובה לכך פשוטה מאוד: **הלקוח**. אין הכוונה לכך שהניהול היומיומי של הפרויקט, שהינו בעיקר פרויקט תוכנה, צריך להיות מנוהל על ידי הלקוח. הכוונה לכך שהלקוח צריך להרגיש "בעלות" על הפרויקט ולהיות מעוניין בהצלחתו.

הדרך לקדם את מעורבות הלקוח הארגוני בביצוע הפרויקט היא לגרום למנהל בכיר מצד המשתמשים לעמוד בראש **ועדת ההיגוי**. ועדת ההיגוי צריכה להיות פעילה ומעודכנת באופן שוטף בהתקדמות בפרויקט. אחת הסוגיות המעניינות היא שבמהלך הקמת מחסן נתונים כלל ארגוני שעשוי להימשך מספר שנים, מתחלפים הנושאים המטופלים ובאופן טבעי גם המשתמשים. לכן, כדאי שבשלב הקמת המקטע השיווקי יעמוד בראש ועדת ההיגוי סמנכ"ל שיווק, ובשלב הקמת מקטע העוסק בניתוחי תקלות בקו הייצור יעמוד סמנכ"ל תפעול או מנהל אבטחת איכות.

המתודולוגיה מבוססת על שלב ראשון של גיבוש תוכנית אב, שבה עדיין לא ברור איזה מקטעים ייושמו ומתי, סביר שבשלב זה יעמוד בראש ועדת ההיגוי סמנכ"ל טכנולוגיות מידע (או מנהל אגף מחשוב, על פי מיקומו בהיררכיה הארגונית). מכאן והלאה יש להעביר את האחריות לידי הלקוחות - מנהלי אגפים בארגון - וכל המקדים הרי זה משובח.

התחל בפרויקט ניסוי (Pilot Project)

הדרך הטובה ביותר להתחיל את המסע להקמת מחסן הנתונים היא על ידי ביצוע פרויקט ראשוני בעל היקף ויעדים מוגבלים יחסית. רצוי שפרויקט ראשוני זה יתבסס על אוסף הנתונים הקריטיים ביותר לארגון, ולא על נתונים בעלי רמת חשיבות נמוכה. צמצום היקף הפרויקט הראשוני יפשט ויקצר מאוד את זמן הקמתו ובאופן טבעי - ישפר את סיכויי הצלחתו. צמצום היקף הפרויקט יקטין גם את התועלות הצפויות ממנו, אולם תיאום ציפיות עם ההנהלה והמשתמשים העתידיים והדרכה נלווית יכולים להבטיח שהמטרות ברורות ולא תהיינה אכזבות מאי מימוש ציפיות.

פרויקט ארגוני

חשוב להתייחס אל פרויקט הקמת מחסן הנתונים כאל פרויקט ארגוני הדורש משאבים מיוחדים ותשומת לב ניהולית שוטפת. למעט מקרים נדירים של הקמת מרכול נתונים קטן, פרויקט הקמת מחסן הנתונים הינו מורכב, מנצל מתודולוגיה שתלווה אותו לאורך כל מחזור חייו, ומשלב צוות מגוון של בעלי כישורים בתהליך הקמתו ותפעולו.

הקמת ועדת היגוי בראשות משתמש בכיר, מעקב שוטף אחר הפרויקט, שילוב המשתמשים באופן מלא, עלות גבוהה - אלה הם רק חלק מהמשמעויות של פרויקט ארגוני. כמו בכל פרויקט ארגוני מעורבות ההנהלה לאורך כל שלבי הפרויקט הינה חיונית להצלחתו. אל תסכים שהארגון ישלם מס שפתיים בלבד, אלא עמוד על כך שתהיה מעורבות פעילה ושוטפת במהלך הפרויקט.

אל תזלזל במאמץ מיפוי נתונים והעברת הנתונים

אל תזלזל במאמץ הנדרש למיפוי מקורות הנתונים ובמאמץ השוטף הנדרש להזרימם מהמערכות התפעוליות אל מחסן הנתונים. בדרך כלל זהו השלב הקשה והמתסכל ביותר. כי רוב המאמץ נעשה מאחורי הקלעים והמשתמשים יודעים עליו מעט מאוד, והוא גם אינו זוכה להכרה ששלבים אחרים זוכים בה. כל הפעלה של שאילתה חדשה המביאה תועלת עסקית כלשהי, או הכנסה לשימוש של כלי קצה כלשהו, זוכים להכרה מצד המשתמשים והמנהלים. למרות כל האמור לעיל, חלק ניכר מהצלחת פרויקט מחסן הנתונים נובע מביצוע שלב אפור ומייגע זה, של הבאת הנתונים מהמערכות התפעוליות, טיובם וטעינתם למחסן הנתונים.

טפל בבעיות אמינות הנתונים החל מהשלב הראשון

קיימת נטייה טבעית להתעלם מבעיות הקשורות לאמינות הנתונים במערכות התפעוליות, ולנסות לבחור בקיצורי דרך שונים. הדרך הנכונה היא לא להתעלם מבעיות אלו, אלא לנסות להתמודד איתן כבר בשלבי הפרויקט הראשונים. איכות הנתונים הנכנסים למחסן הנתונים קריטית להצלחתו, ולכן יש להבטיח זאת. קיצורי דרך או התעלמות מאיכות הנתונים יכולים להביא לטעויות חמורות בניסיון להסיק מסקנות כלשהן מהפרויקט הראשוני לגבי הצפוי בתחילת היישום האמיתי. לצערי, לעיתים לא ניתן לפתור את בעיות איכות הנתונים, אלא על ידי ביצוע שינויים במערכות התפעוליות, דבר ארגונים רבים מעדיפים להימנע ממנו. עם זאת, המסקנה היא שצריכים להיות מודעים לכך כבר בהתחלה, ולא לדחות את הטיפול לשלבים מאוחרים יותר.

הייה מציאותי בכל הקשור לבעיות סנכרון הנתונים בתוך מחסן הנתונים

כפילות נתונים היא תמיד מקור לבעיות. הבעיה היא שתפיסת מחסן הנתונים מבוססת על כפילות נתונים: כפילות הנובעת מניהול כפול של נתונים במערכות התפעוליות ובמחסן הנתונים, וכפילות נתונים המנוהלת במחסן הנתונים עצמו מתוך מטרה לשפר את ביצועיו (למשל ניהול של נתונים מסוכמים). במשך שנים למדנו עד כמה כפילות נתונים יכולה להזיק ומהי רמת הסיכון בניהול נתונים כפולים, גם למדנו טכניקות שונות שמטרתן העיקרית היתה מניעת כפילות נתונים. והנה אנו צריכים להכיר ב"עובדת החיים" של כפילות הנתונים באמצעות מחסן הנתונים.

כפילות נתונים שנכפית עלינו בעולם מחסן הנתונים מחייבת חשיבה ותכנון מדוקדקים מראש, כדי למנוע ולמזער את הבעיות האפשריות. אחת הבעיות הכאובות ביותר היא מציאת אי התאמות בדיווחים שונים שמקורם במערכות התפעוליות לעומת דיווחי מחסן הנתונים. בעיה זו גורמת לאיבוד אמון המשתמשים באמיתות הנתונים המנוהלים במחסן הנתונים. המאמץ לשמור על סינכרון בין המערכות התפעוליות לבין מחסן הנתונים הינו מאמץ בלתי פוסק ומקור לאכזבות וקשיים. יש להתייחס לנושא זה ברצינות הראויה ולבנות מראש מנגנוני בקרה והשוואה, שיאפשרו לאתר מוקדם ככל הניתן את אי ההתאמות ולטפל בהן.

בחר בתבונה את כלי הקצה

בסופו של דבר, הכלים העומדים לרשות משתמשי הקצה הם החלון שדרכו המשתמשים מתבוננים ומנתחים את הנתונים המנוהלים במחסן הנתונים. מומלץ להקדיש זמן ללימוד ולהערכת כלי הקצה השונים, הן מנקודת מבט המשתמש והן מנקודת מבט אנשי המחשוב (תפעול הכלי, בטיחות נתונים, ניהול שוטף וכדומה).

בנה הערכה ריאלית של כוח האדם הנדרש

מחסן הנתונים הינו תוצר של עבודת צוות ממספר דיסציפלינות מחשוב שונות: משתמשים, מנהלי בסיסי נתונים, מנתחי מערכות, מומחים לכלי קצה שונים, מפתחי תוכנה, אנשי תפעול וכדומה. קח זאת בחשבון ואל תניח שמחסן הנתונים **שאתה** עומד להקים לא ידרוש משאבים אלה. לעיתים קורים ניסים, אבל לא בכל הקשור להקמת ולתפעול של מחסני נתונים. ה"ניסים" קורים ופועלים רק בזכות המאמץ והדבקות במטרה של צוות הפרויקט לאורך זמן.

הגדר את כללי אבטחת המידע

מחסן הנתונים מכיל מידע רגיש וחשוב ולכן יש לתת את הדעת על ההרשאות שתינתנה למשתמשים במטרה לוודא שכללי אבטחת מידע יקוימו. במערכות התפעוליות ההגנה על הנתונים מתבצעת על ידי היישומים עצמם. לעומת זאת, במחסן הנתונים יש למשתמש חופש פעולה גדול יותר והוא ניגש באופן ישיר לנתונים ללא יישומים מתווכים. כלי הקצה השונים ובסיס הנתונים עצמו מספקים רמות שונות של הרשאות, שבאמצעותם ניתן לפתור חלק גדול מבעיות אבטחת המידע.

צפה שעלויות הקמת מחסן הנתונים תהיינה גבוהות מהמתוכנן

פרויקט מחסן הנתונים אינו פרויקט זול, גם אם מיישמים מרכול נתונים על שרת קטן יחסית. מעבר למשפטים המרגיעים שספקי הכלים משמיעים ומעבר למתודולוגיות האינסטנט השונות, כל הקמה של מחסן נתונים ותפעולו היא **אתגר** של ממש. יש לזכור ש"התאבון בא עם האכילה", ולכן יישום מוצלח בשלבים הראשונים מעורר מייד דרישות חדשות שלא נצפו מראש, שמאחוריהן מסתתרות עלויות חומרה ותוכנה ומאמצי יישום חדשים, וחוזר חלילה. כלומר, ההצלחה היא שסוחפת את גידול העלויות.

טבעם של מחסני הנתונים לגדול ולגדול, ומי שהתנסה בניהול בסיס נתונים גדול יודע את האתגר המיוחד הזה. עלויות החומרה, הרשיונות של כלי הקצה, עלויות כוח אדם תומך - כל אלה הם רק חלק מהעלויות. חלק מהעלויות מוסתר בצוותי תוכנה של המערכות התפעוליות המחלצים את הנתונים מהמערכות התפעוליות, ועוקבים אחר הדרישות המשתנות מדי פעם, כדי שהמשתמשים יוכלו לעקוב אחר השינויים בתהליכים העסקיים של הארגון.

בנה קטלוג נתונים טוב

חלק גדול מהצלחת פרויקט מחסן נתונים נובע מקטלוג נתונים טוב המאפשר למשתמשים להתמצא בנבכי מחסן הנתונים, להבין את משמעות הנתונים, להבין את רמת עדכניותם, מהיכן נגזרו וכדומה. חשוב לעשות מאמץ ארגוני וכספי כדי לבנות קטלוג נתונים ראוי, למרות שהכלים הקיימים כיום לניהול קטלוג נתונים הם מוגבלים יחסית ואינם מסוגלים לכסות את כל הספקטרום של סוגי המידע שיש לנהל בקטלוג.

למד את המשתמשים המתוחכמים ואת מנתחי המידע כיצד לבנות שאילתות טובות

פרויקט מחסן הנתונים מיועד לאפשר למשתמשים לפעול באופן עצמאי באמצעות כלי השאילתות והניתוח. היכולת של המשתמשים להפעיל את הכלים באופן עצמאי וללא תיווך יחידת המחשב הינו גורם חשוב בהצלחת הפרויקט. על כן, אל תחסוך במאמץ להדרכת המשתמשים. צריך לאפשר למשתמשים להציג בעצמם את דרישותיהם למידע באמצעות שאילתות שהם ינסחו, ולא באמצעות שאילתות שיפותחו עבורם מראש על ידי אנשי תוכנה.

העוצמה של מחסן הנתונים נובעת מתהליכי גילוי הנתונים והמידע, תהליכים שיכולים להתרחש רק אם המשתמשים המבינים את משמעות הנתונים יתחילו לחקור את הנתונים בעצמם. מכיון שהמשתמשים לא בהכרח יודעים כיצד כלי השאילתות פועלים, הם יכולים לבנות בקלות שאילתות לא יעילות שזמן התגובה שלהם בלתי נסבל ומבזבז משאבי מחשב יקרים והתוצאה - אי שביעות רצון. חשוב להסביר למשתמשים כיצד בונים שאילתות יעילות ולאפשר להם לבחון את ביצועי השאילתות שהם בונים.

סיכום

פרק זה עסק במתודולוגיה להקמת מחסני נתונים. כפי שראינו, מחזור החיים של מחסן נתונים שונה ממחזור חיים של מערכת מידע רגילה. מחזור החיים שהוצג שם דגש מיוחד על שני השלמים הראשונים, שבדרך כלל הארגונים נוטים להזניחם: בחינת הצורך העסקי וגיבוש ארכיטקטורת-על. יש חשיבות רבה לכך שמחסן הנתונים יהיה חלק מאסטרטגיית המידע של הארגון וישתלב בתמונה הכוללת.

אני מניח שעכשיו, לאחר שהקורא סיים את קריאת הספר, ברור לו שהקמת מחסן נתונים היא תהליך מורכב, יקר ולעיתים ארוך. מכל הטעמים האלה נובע שארגון שמתחיל ביישום מחסן נתונים צריך לשאול את עצמו מספר שאלות, כגון: מהן מטרות המערכת החדשה, אילו תועלות היא תביא, מהם הנושאים שתטפל בהם, מהם סדרי העדיפות ליישום, מי הם המשתמשים העתידיים במערכת, מהי העלות להקמה ותחזוקה, איזה משאבים דרושים, ועוד. ארגון שלא ישאל את השאלות הראויות ולא ינתח את התשובות המתקבלות, מסתכן בכניסה לפרויקט שסיכויי הצלחתו נמוכים.

התועלת והחשיבות של סביבת מחסן הנתונים לארגון הם כה רבים, שאסור לארגון הסתכן בכישלון. השהיית תהליך ההקמה של מחסן הנתונים, או ביטולו, עלולה לגרום אבדן זמן יקר שאינו בנמצא עבור ארגון הפועל בסביבה תחרותית, ואפילו לגרום לארגון נזקים גדולים.

לפיכך נחזור על דברינו: **בדוק, החלט, תכנן והחל בתהליך היישום.**

נספח א': מילון מונחים

מילון המונחים מציג באופן תמציתי את המונחים העיקריים שנסקרו בספר זה. המילון מוצג במיון אלפביתי על פי המונח הלועזי. ליד כל מונח לועזי מופיע המונח העברי וכן תיאור תמציתי של המונח.

מונח לועזי	מונח עברי	עמוד	תיאור המונח
Additive Attributes	תכונות ברות חיבור	136 173	תכונות נומריות המנוהלות בטבלת העובדות, שניתן לבצע עליהן פעולות אריתמטיות במיגוון חתכים שונים ולקבל תוצאה בעלת משמעות.
Ad-Hoc Query	שאילתה מזדמנת		שאילתה שלא ניתן להגדירה מראש. שאילתה זו נבנית בדרך כלל על ידי המשתמש בעת הצורך, תוך שימוש במחוללי שאילתות המייצרים פקודות SQL דינמיות.
Aggregate Data	נתוני סיכום	163	נתונים שהם תוצאה של תהליך סיכום כלשהו של נתונים ברמה נמוכה יותר. לדוגמה, סך המכירות למוצר ליום (סיכום של כל הסניפים).
Aggregate Table	טבלת סיכום	164	טבלה המכילה את נתוני הסיכום המחושבים מראש, ולא תוך כדי ביצוע שאילתה. טבלאות אלו אופייניות מאוד בסביבת מחסן הנתונים ומאפשרות ביצועים טובים עבור שאילתות הזקוקות למידע סיכומי. טבלאות אלו מיוצרות בדרך כלל בתום תהליך טעינת נתונים חדשים למחסן הנתונים. למרות שהן מכניסות מימד של כפילות נתונים ומסבכות את ניהול בסיס הנתונים, חשיבותן לשיפור הביצועים רבה.
Analytical Process	תהליך אנליטי	69 71 80	תהליך שמטרתו לנתח ולהבין את הנתונים. מקובל להבחין בין רמות מורכבות שונות של תהליכים אנליטיים. מאחר וקיים אוסף שלם של תהליכי וכלי ניתוח מקובל לקרוא להם גם המרחב האנליטי.

מונח לועזי	מונח עברי	עמוד	תיאור המונח
API	ממשק תכנות		ראה Application Programming Interface
Application Programming Interface (API)	ממשק תכנות	237	ממשק ידוע וציבורי המגדיר כיצד ניתן להפעיל פונקציות מסוימות של מערכת תוכנה כלשהי על ידי תוכנה חיצונית כלשהי. הממשק מגדיר אילו פונקציות התוכנה יודעת לבצע וכיצד מבוצעות פונקציות אלו, ובכלל זה הגדרה מדויקת של כל הפרמטרים שיש להעביר להן ואיזה פרמטרים הן מחזירות. לדוגמה, ODBC הינו ממשק תכנות לגישה לבסיסי נתונים שונים. דוגמה נוספת היא ממשקי התכנות שרוב יצרני OLAP מספקים כדי לאפשר לתוכנות צד שלישי לגשת אל הנתונים הרב-מימדיים.
Architecture	ארכיטקטורה		הגדרה ועיצוב ראשוני המתאר את מרכיבי הפתרון וכיצד הם פועלים ביחד. ניתן להתייחס לארכיטקטורה כאל תוכנית אב שעל פיה המיישמים יכולים לבנות מערכת העונה על דרישות המשתמש.
Atomic Data	נתונים אטומיים	151	נתונים המייצגים את רמת הפירוט הנמוכה ביותר במחסן הנתונים. לדוגמה, המכירות בסניף לפי מוצר ביום כלשהו. במערכות תפעוליות ניתן למצוא כמובן, רמות פירוט נמוכות יותר.
Attribute	תכונה	84 136 260	יחידת המידע הבסיסית ביותר של יישות (Entity). דוגמאות לתכונות של יישות עובד הן מספר זהות, שם עובד, כתובת, תאריך לידה, תואר אקדמי אחרון וכד׳.
Backpropagation Neural Network	רשת עצבית עם השלכה לאחור	193	מבנה של רשת עצבית שבה קיימים קשרים בין שכבה אחת לבין אחת או יותר מהשכבות הקודמות. רשתות מסוג זה מווסתות את עצמן ו ״לומדות״ מתוך הנסיון.
Bitmapped Index	אינדקס מפת סיביות		מבנה של אינדקס נפוץ במערכות RDBMS המאפשר גישה מהירה מאד אל הנתונים. בחלק מהמקרים אינדקסים מסוג זה יכולים למנוע לחלוטין את הצורך בסריקת טבלאות ואפילו ביצוע צירוף בין טבלאות.
Bulk Load	טעינה המונית	51	טעינה של כמות גדולה של נתונים בבת אחת. זוהי שיטת הטעינה המקובלת של נתונים למחסן הנתונים.
Business Transaction	תנועה עסקית		ראה Transaction.

תיאור המונח	עמוד	מונח עברי	מונח לועזי
ראה Repository		קטלוג	Catalog
היחידה הבסיסית ביותר במערכת OLAP. כל תא יכול להכיל נתון או נוסחה (בדומה לגיליון אלקטרוני) במפגש בין כל המימדים המנוהלים. מערכות OLAP צריכות לדעת לטפל במספר רב של תאים ריקים (Sparse) כי לא בכל מפגש בין המימדים יש נתון.	90	תא	Cell
יישום של תפיסת מחסן נתונים שבה מחסן אחד מרכזי משמש את כל צרכי הארגון. כלומר, המחסן המרכזי משרת את כל היחידות העסקיות של הארגון. ראה גם Enterprise Data Warehouse.	298	מחסן נתונים מרכזי / ארגוני	Centralized Data Warehouse
איתור וניהול השינויים במערכות התפעוליות ושיטת ייצוג שינויים אלה במחסן הנתונים.	158 238	ניהול שינויים	Change Management
תהליך המזהה את הנתונים במערכות התפעוליות שחלו בהם שינויים מאז אחזור הנתונים האחרון. תהליכים אלה מבוססים בדרך כלל על תוכנה מיוחדת המכירה את מבנה יומן האירועים (Log File) של המערכת לניהול בסיס הנתונים ומסוגלת לנתח אותו על מנת לאתר את השינויים בלבד.	239	זיהוי שינויים בנתונים	Changed Data Capture
אלגוריתם המבוסס על מבחן Chi בריבוע המופעל על אוסף של נתונים ומייצר את הענפים בתוך עץ החלטות.	192 197	אלגוריתם חי בריבוע לבניית עצי החלטות	Chi Square Automatic Interaction Detection (CHAID)
טכניקת ניתוח נתונים נפוצה בתחום כריית הנתונים ומשמשת בעיקר לתהליכי חיזוי. הטכניקה מאפשרת לבנות מודל סיווג כלשהו מתוך בסיס הנתונים היסטורי. בדרך כלל טכניקת הסיווג משתמשת בהקבצות ידועות מתוך הנתונים כדי לפתח את שיטת הסיווג, או לחילופין מפתחת הקבצות חדשות ובלתי ידועות מתוך הנתונים. השיטה מנסה לשייך להקבצה אחת את האלמנטים "הקרובים" ביותר ביניהם. לדוגמא בעיית סיווג נפוצה היא חלוקת בסיס נתוני הלקוחות להקבצות הומוגניות ככל הניתן בהקשר של סיכון מתן אשראי.	189	סיווג	Classification
טכנולוגיה מחשובית המבוססת על ביזור של היישום בין מספר שכבות חומרה ותוכנה. הביזור הוא בדרך כלל על פי התפקוד המיוחד - תצוגה	32	שרת/לקוח	Client/Server

מונח לועזי	מונח עברי	עמוד	תיאור המונח
			וניווט, ניהול היישום וניהול בסיס הנתונים. מחשב הלקוח עוסק בדרך כלל בתצוגה בשעה שמחשב השרת עוסק בפונקציות המשותפות למספר לקוחות, כגון לוגיקה עסקית משותפת, ניהול בסיס נתונים וכד׳. מחסני הנתונים מבוססים במידה רבה על טכנולוגיה זו, הן בגלל יכולה להציג את הנתונים באופן גרפי ומתוחכם והן בגלל יכולתה לנהל כמויות נתונים גדולות בשרתים מרובי מעבדים, בעלויות סבירות.
Cluster Analysis	ניתוח אשכולות	189	טכניקות ניתוח נתונים העוסקות בפילוח אוכלוסיות גדולות של אובייקטים לאוסף כלשהו של אשכולות בעלי הגיון פנימי כלשהו. כלומר, קיים דמיון מסוים בין האובייקטים השייכים לאשכול בהתבסס על קריטריון כלשהו.
Computer Aided Software Engineering (CASE)	הנדסת תוכנה נתמכת מחשוב	257	הנדסת תוכנה נתמכת כלים ממוחשבים.
Copy Management	ניהול העתקות		רכיב תוכנה מיוחד וייעודי העוסק בניהול העתקות של קבצים מסוגים שונים בין סביבות מחשוב שונות והטרוגניות. מקובל לקרוא בשם כוללני זה לכל הכלים בסביבת מחסן הנתונים העוסקים בהעתקת והעברת הנתונים ממערכות תפעוליות אל סביבת מחסן הנתונים.
Cross Selling	מכירה צולבת		גישה עסקית המציעה ללקוח מוצרים שונים תוך כדי רכישת מוצר כלשהו. לדוגמא, ניתן להציע לכל לקוח בעל חשבון עו״ש מעל סכום מסויים, לרכוש גם כרטיס אשראי.
Cube	קוביה	89	מבנה מיוחד המנהל את הנתונים על פי מימדים עסקיים, כמו מוצרים, סניפים, זמן, לקוחות וכד׳. הקוביה הרב-מימדית הינה המטאפורה הנפוצה ביותר לתיאור מבנה הנתונים המנוהל במערכות OLAP.
Customer Relationship Management (CRM)	ניהול קשרי לקוח		אוסף של מערכות מידע וטכנולוגיות מחשוב העוסקות בניהול הקשרים של הלקוח עם הארגון. מקובל לשייך לקבוצה זו מערכות המטפלות במוקדי שרות, ניהול תקלות, טלמרקטינג, דיוור ישיר וכד׳.

מונח לועזי	מונח עברי	עמוד	תיאור המונח
Daily Update Window	חלון עדכון יומי		פרק הזמן במשך יום אחד שבו ניתן לבצע את הטעינות של הנתונים החדשים אל מחסן הנתונים. לדוגמא אם הגזירה של הנתונים מהמערכות התפעוליות מסתיימת בשעה 24:00 בלילה, אזי חלון העדכון היומי הוא 8 שעות, בהנחה שמחסן הנתונים צריך להתחיל לפעול בשעה 8:00 בבוקר.
Data	נתונים	65	אוסף של עובדות גולמיות על יישויות שונות. עובדות אלו יכולות להיות מיוצגות על ידי מספרים, מלל, תמונות, קול וכד׳. הנתונים נוצרים בדרך כלל על ידי המשתמשים המזינים את המערכות התפעוליות כתוצאה מעבודתם השוטפת. לדוגמה, כתובת סטודנט, מחיר פריט, מספר סניף וכד׳. זוהי היחידה הבסיסית ביותר הנשמרת במחסן הנתונים.
Data Access Tools	כלי גישה לנתונים	54	כלים המיועדים למשתמשי קצה שבאמצעותם הם יכולים לתחקר, לנתח ולהציג את הנתונים. סביבת מחסן הנתונים עתירה בכלי גישה לנתונים רבים, כל אחד המתאים לסוג משתמשים מסוים, או לפעולות ניתוח מידע. דוגמאות לכלי קצה הם מחוללי דוחות, מחוללי שאילתות, כלים סטטיסטיים, כלי OLAP, גיליון אלקטרוני וכד׳.
Data Analysis	ניתוח נתונים	63 70	ענף של סטטיסטיקה העוסק בניתוח אוספים גדולים של נתונים.
Data Cleansing	טיוב נתונים	245	תהליך העוסק בשיפור וטיוב הנתונים לקראת טעינתם למחסן הנתונים. התהליך עוסק בשינוי פורמטים, בדיקות תקינות, הסבת קודים וכד׳.
Data Denormalization	דה-נורמליזציה של נתונים	131 144 251 300	תהליך עיצוב מיוחד המרשה הכנסת כפילויות של נתונים אל מודל הנתונים, מתוך מטרה לבנות מבנה פשוט יותר, קל יותר להבנה ובדרך כלל גם יעיל יותר מאחר ואין צורך בצירוף (Join) של טבלאות רבות. תהליך זה מקובל בסביבות מחסני נתונים, ומאוד לא פופולרי ומומלץ לסביבות תפעוליות.
Data Dictionary	מילון נתונים		ראה Repository
Data Element	נתון		היחידה הבסיסית ביותר של הנתונים, כמו שם עובד, כתובת סניף, מחיר פריט וכד׳. חלוקה נוספת של הנתון בדרך כלל אינה בעל ערך עסקי מבחינת הארגון.
Data Extracting	גזירת נתונים	233 235	תהליך העוסק בגזירת נתונים מהמערכות התפעוליות.

מונח לועזי	מונח עברי	עמוד	תיאור המונח
Data Extraction Software	תוכנה לגזירת נתונים	238	תוכנה ייעודית שמטרתה לאפשר חילוץ נתונים ממערכות תפעוליות כדי להעבירם לסביבת מחסן הנתונים. תוכנות אלו מתמחות בקריאת מיגוון מבני נתונים ובסיסי נתונים המנוהלים בסביבות מחשוב שונות. הן גם יכולות להפעיל לוגיקה לבחירת הנתונים לחילוץ ולטיוב והמרת הנתונים.
Data Loading	טעינת נתונים	227 234 253	תהליך הטוען את הנתונים אל בסיס הנתונים של מחסן הנתונים. הנתונים הנטענים הם לאחר תהליכי טיוב והמרה. תהליך זה הינו בעל חשיבות רבה בסביבת מחסן הנתונים, כי זו השיטה הנפוצה ביותר לעדכון מחסן הנתונים. חלון הזמן לטעינת הנתונים מוגבל בדרך כלל, ולכן תוכנות הטעינה מפעילות שיטות עבודה במקביל.
Data Management	ניהול נתונים		ענף של טכנולוגיית המידע העוסק במיגוון פעילויות הקשורות לניהול הנתונים: אחסון, חילוץ (Extraction) מהיר של נתונים באמצעות אינדקסים, בקרת גישות לנתונים, ניהול אמינות ושלמות הנתונים בעת ביצוע תנועות, ועוד.
Data Management Software	תוכנה לניהול נתונים		תוכנה המבצעת את כל פעולות ניהול הנתונים. אלו הן המערכות לניהול קבצים והמערכות לניהול בסיסי הנתונים.
Data Mapping	מיפוי נתונים	265	תהליך המאפשר מיפוי נתונים בין מקור ליעד כלשהו. מקובל להשתמש במונח מיפוי נתונים לתהליך המזהה את כל הנתונים במערכות התפעוליות - המקור - ומגדיר לכל נתון להיכן וכיצד הוא מועבר אל מחסן הנתונים - היעד. תהליך זה הינו אחד התהליכים הבסיסיים ביותר בעת עיצוב מחסן הנתונים.
Data Mart	מרכול נתונים	56 269	המרכול הוא נגזרת מסוימת של מחסן נתונים ארגוני. מרכול הנתונים מוגבל בדרך כלל לנושא מסוים או למבנה מיוחד הנדרש על ידי כלי גישה לנתונים כלשהו, ומשמש בדרך כלל אוכלוסיה מצומצמת של משתמשים.
Data Mart Architecture	ארכיטקטורה של מרכול נתונים	56 273	הארכיטקטורה הכוללת של מרכול הנתונים - כלומר כיצד הוא בנוי, מהן השכבות השונות המרכיבות אותו, מהן הטכנולוגיות בהן הוא משתמש וכד׳.
Data Mining	כריית נתונים	76 179	תהליך ניתוח נתונים המבוסס על טכנולוגיות תוכנה מתקדמות שמטרתו למצוא מידע חדש

מונח לועזי	מונח עברי	עמוד	תיאור המונח
			ובלתי ידוע מתוך מחסן הנתונים. מידע זה בדרך כלל לא ידוע למשתמש ועוסק בדרך כלל במציאת תבניות, מגמות מיוחדות, חוקיות בתוך הנתונים, הקבצות של הנתונים לקבוצות בעלות משמעות לוגית כלשהי וכד׳. טכנולוגיות התוכנה המשמשות את כריית הנתונים מהוות שילוב כלשהו של כלים סטטיסטיים וכלים שמקורם באינטליגנציה מלאכותית ומערכות לומדות.
Data Model	מודל הנתונים	298	מודל לוגי המייצג את היישויות, את התכונות של כל יישות ואת הקשרים בין היישויות. זהו מודל לוגי מבחינת אי-התלות שלו במערכת חומרה או תוכנה כלשהי. כך הוא מהווה ייצוג רעיוני (קונצפטואלי) של הנתונים.
Data Modeling	עיצוב מודל הנתונים	298	תהליך העוסק בהקמת מודל הנתונים. תהליך זה מבוצע בדרך כלל על ידי מנתחי מערכות או על ידי מנתחי נתונים שהוכשרו והתמחו בעיצוב מודלים של נתונים. זהו אחד התהליכים החשובים ביותר בסביבת מחסן הנתונים כי הוא עוסק בבניית מודל הנתונים שיעמוד לרשות מקבלי ההחלטות ולרשות כלי השאילתות השונים.
Data Normalization	נירמול נתונים	126 129	תהליך שמטרתו לבנות מבנה נתונים עם מינימום כפילות נתונים. שיטת עיצוב מקובלת לעיצוב בסיסי נתונים תפעוליים.
Data Partitioning	חלוקת נתונים למחיצות	205 218	תהליך העוסק בחלוקת טבלה לוגית אחת למספר טבלאות פיסיות או לוגיות. מטרת התהליך בדרך כלל לשפר ביצועים בסביבות מחשוב מרובות מעבדים התומכות בעיבוד מקבילי של מחיצות שונות. בנוסף לשיפור בביצועים משמש תהליך החלוקה למחיצות גם לניהול קל יותר של טבלאות גדולות.
Data Pivot	ציר מרכזי	97	שיטה המקובלת במיוחד בכלי OLAP המאפשרת התבוננות על נתון כלשהו מנקודות מבט שונות. מבטים אלה יכולים להיות למשל, הצגת המכירות לפי מימד המוצר, מימד הזמן או מימד הסניף וכד׳. מקובל גם לקרוא לתהליך זה בשם סיבוב מימדים.
Data Propagation	הפצת נתונים	238	תהליך העוסק בהפצת הנתונים מסביבות המקור שלהם, המערכות התפעוליות, ועד למערכת היעד שלהם - מחסן הנתונים. בדרך כלל, תהליך זה נתמך על ידי תוכנות החילוץ, או תוכנות ייעודיות,

מונח לועזי	מונח עברי	עמוד	תיאור המונח
			המסוגלות להפעיל כללי הפצה מתוחכמים (הפצה כל יום, או כשהצטברו שינויים בכמות מסוימת).
Data Quality	איכות נתונים		המידה שבה הנתונים מייצגים באופן אמין ושלם את המציאות הרלוונטית.
Data Replication	שכפול נתונים	58	טכניקה מיוחדת המאפשרת שכפול טבלאות או חלקי טבלאות ממקור כלשהו אל יעד אחד או יותר. מערכות התוכנה התומכות בשכפול נתונים הן מורכבות ומסוגלות לזהות את השינויים ולהפיץ את הנתונים אל היעדים השונים. הן עושות כל זאת תוך אבטחת רמה גבוהה של הצלחה.
Data Source	מקור נתונים	54 299	מקור של נתונים עבור מחסן הנתונים. מקובל להבחין בין מקור שנובע מהמערכות התפעוליות של הארגון לבין מקור נתונים חיצוני.
Data Staging	אחסון נתונים	54	שטח אחסון ביניים שבו מאחסנים את הנתונים לקראת טעינתם למחסן הנתונים.
Data Store	מאגר נתונים		שם כוללני למקום שבו מאוחסנים נתונים, בין אם במבנה פשוט של קבצים שטוחים, או במבנה מורכב יותר של בסיס נתונים המנהל גם את הקשרים בין הנתונים. בתהליך העברת הנתונים מהמערכות התפעוליות אל מחסן הנתונים, מקובל להשתמש במספר מאגרי ביניים בהם מאחסנים את הנתונים תוך כדי תהליך הטיוב וההמרה ועד לטעינתם למחסן הנתונים.
Data Transformation	המרת נתונים	54 234 248	תהליך העוסק בהמרת הנתונים ממערכות המקור אל המבנה המיוחד של מחסן הנתונים, תוך אבטחת רמה גבוהה של איכות ותקינות הנתונים. מקובל לשייך לתהליכים אלה פעולות של בדיקת תקינות, הוספת נתונים חסרים, המרת קודים למונחים עסקיים, פתרון של סתירות פנימיות בתוך הנתונים, שינוי פורמט הייצוג של הנתונים לפורמט אחיד המנוהל במחסן הנתונים, המרת מבנה הנתונים למבנה הנדרש על ידי מחסן הנתונים וכד׳.
Data Type	טיפוס נתונים	146	תכונה המגדירה את טיפוס הנתונים: נומרי, אלפאנומרי, מחרוזת סיביות וכד׳.
Data Warehouse	מחסן נתונים	46 50	סביבת מחשוב מיוחדת לניהול נתונים המיועדים לתמיכה בתהליכי קבלת החלטות. סביבה זו מבוססת על מיגוון טכנולוגיות מחשוב מודרניות המסוגלות לנהל נפחי נתונים גדולים ולהעמיד את הנתונים לרשות קהל מגוון של מקבלי החלטות.

תיאור המונח	עמוד	מונח עברי	מונח לועזי
ראה גם את ההגדרה של מחסן הנתונים הארגוני - Enterprise Data Warehouse.			
תמונה כוללת של מיגוון תהליכי המיפוי, הטיוב וההמרה שהנתונים עוברים ושל כל טכנולוגיות המחשוב המשמשות לניהול סביבת מחסן הנתונים.	52 287	ארכיטקטורת מחסן הנתונים	Data Warehouse Architecture
מקטע אחד מתוך מכלול הנושאים בהם מטפל מחסן הנתונים. מקובל לחלק את יישום מחסן הנתונים לאוסף מקטעים המיושמים באופן מדורג על ציר הזמן. ניתן להתייחס למקטע כאל תת-פרויקט שיש לו התחלה וסוף, ועוסק בקטע מסוים של מחסן הנתונים.	287 305	מקטע של מחסן נתונים	Data Warehouse Increment
אוסף של כלי התשתית המשמשים את סביבת מחסן הנתונים: חומרה, בסיס הנתונים, כלי הגזירה, העתקה, טיוב והמרה, וסביבת התקשורת.	301	תשתית מחסן הנתונים	Data Warehouse Infrastructure
אוסף מובנה של צעדים, עקרונות, תהליכים ופעולות שיש לבצע כדי לבנות וליישם מחסן נתונים. קיימות מתודלוגיות שונות, אולם רמת הדמיון ביניהן היא רבה.	283	מתודולוגיה ליישום מחסן נתונים	Data Warehouse Methodology
בעל תפקיד מיוחד ביחידת המחשב המתמחה בניהול שוטף של בסיס הנתונים על כל מיגוון הפעילויות הנדרשות. פעילויות אלו יכולות להיות: הגדרת בסיס הנתונים, בניית והורדת אינדקסים, שינויים בטבלאות, מעקב אחר ביצועים, הגדרת הרשאות, כוונון שוטף של בסיס הנתונים לשיפור הביצועים, גיבוי ושחזור בסיס הנתונים, וכד׳. בסביבת מחסן הנתונים נדרשת התמחות מיוחדת מעבר להתמחות הרגילה בנושאי ניהול בסיסי נתונים גדולים, חלוקת בסיס הנתונים למחיצות וכד׳.	303	מנהל בסיס הנתונים	Database Administrator (DBA)
ההגדרה הלוגית והפיסית של בסיס הנתונים ובכלל זה הגדרת הטבלאות המרכיבות אותו, העמודות של כל טבלה, המפתחות של כל טבלה, כללי האמינות המיוחדים לכל טבלה, ההרשאות לכל טבלה, מבנה הטבלאות המדומות (Views) ועוד.		סכמת בסיס הנתונים	Database Schema
ראה Data Base Administrator.		מנהל בסיס הנתונים	DBA
מערכת ייעודית המנהלת נתונים הנדרשים לקבלת החלטות ומכילה אוסף של כלים לתחקור, הצגת וניתוח הנתונים.	31 43	מערכת תומכת החלטות	Decision Support Systems (DSS)

תיאור המונח	עמוד	מונח עברי	מונח לועזי
טכניקת ניתוח נתונים המאפשרת לזהות גורמים משפיעים ולמיין אותם על פי סדר החשיבות שלהם. הצגת תוצאות הניתוח היא במבנה עץ היוצא משורש, מסתעף למספר ענפים ומסתיים בעלים.	77 190	עצי החלטה	Decision Trees
מימד המופיע בטבלת העובדות אולם מאחוריו אין טבלת מימד.	148	מימד מנוון	Degenerate Dimension
אוסף של נתונים המזהים את התכונות של וכלוסיה מסוימת. לדוגמא גיל, אזור מגורים, רמת הכנסה, ארץ לידה , מין, גודל משפחה וכד׳.	149	נתונים דמוגרפיים	Demographic Data
נתונים המהווים תוצאה של תהליך חישוב כלשהו על נתונים אחרים. מקובל להשתמש בנתונים תוצאתיים בעיקר בתהליכי סיכום או במצבים שבהם מבקשים לחסוך מקום אחסון (למשל, את ערך סה״כ המכירה ניתן לחשב על ידי כפל הכמות במחיר הפריט).	92	נתונים תוצאתיים	Derived Data
מערכת OLAP הפועלת על מחשב אישי. אלו הן מערכות פשוטות וזולות יחסית, הטוענות את הנתונים אל בסיס נתונים ייעודי המנוהל במחשב האישי. הביצועים בדרך כלל טובים, אולם היקף הנתונים מצומצם.	109	מערכת רב-מימדית שולחנית	Desktop OLAP (DOLAP)
מונח בעל משמעות עסקית בעולמו של המשתמש ומיועד למדידת ביצועי העסק. דוגמאות : לקוחות, מוצרים, סניפים, זמן, מבצעים וכד׳.	86 137	מימד	Dimension
לכל מימד יש מספר תכונות. לדוגמה, למימד סניף יש תכונות כגון שם הסניף, כתובת הסניף, שם האזור אליו שייך, שטח הרצפה של הסניף וכד׳. תכונות אלו מאפשרות למשתמשים להגדיר את החתכים השונים לקבלת המידע.	137	תכונות המימד	Dimension Attributes
בדרך כלל למימדים יש היררכיות הקושרות רמות שונות של המימד, והן בעלות משמעות בעולמו של המשתמש. לדוגמה, בהיררכיה של הזמן יש יום, שבוע, חודש, רבעון, שנה. במימד הסניפים יש סניף, אזור, מחוז וכד׳.	87 144	היררכיית מימד	Dimension Hierarchy
טבלה המהווה חלק מסכמת כוכב ומנהלת את הנתונים הרלוונטיים לכל מימד.		טבלת מימדים	Dimension Table
תהליך עיצוב מיוחד לסביבות של מחסני נתונים המיועד לבניית מודל נתונים פשוט להבנה ויעיל מבחינת הביצועים. מודל זה עושה אבחנה בין שני	125	עיצוב מימדי של מודל הנתונים	Dimensional Data Modeling

תיאור המונח	עמוד	מונח עברי	מונח לועזי
סוגים של טבלאות - טבלאות של עובדות וטבלאות של מימדים ובאוסף קשרים מיוחדים ביניהם באמצעות מפתחות ייעודים.			
שיטת ניתוח נתונים שבה כלי תוכנה כלשהו מנסה לגלות מידע חדש מתוך הנתונים, בדרך כלל ללא הנחות כלשהן מראש. רוב כלי כריית הנתונים מבוססים על מודל זה.	73 181	ניתוח מבוסס גילוי	Discovery Based Analysis
ראה Desktop OLAP.		מערכת רב-מימדית שולחנית	DOLAP
אוסף תהליכים המטפלים בנתונים שמתיישנים והורדתם להיסטוריה ולאמצעי אחסון משניים.	232 255	תהליכי הזרימה מטה	Down Flow Process
פעולה מקובלת במערכות OLAP המאפשרת לקבל פירוט נוסף. למשל, מהצגת סיכום המכירות למוצר אפשר לעבור לקבלת סיכום המכירות למוצר לפי חודש.	76 97 99	קידוח מטה	Drill Down
פעולה הפוכה לקידוח מטה. כלומר, מעבר מרמה מפורטת לרמת סיכום גבוהה יותר.	101	קידוח מעלה	Drill Up
ראה Decision Support Systems.		מערכת תומכת החלטות	DSS
ראה Executive Information Systems.		מערכות מידע למנהלים	EIS
משתמש של מערכת מידע כלשהי. בסביבת מחסן הנתונים, משתמש הקצה הינו העובד המבקש לגשת לנתונים, לנתח אותם ולהשתמש בהם לקבלת החלטות.	297	משתמש קצה	End User
זיהוי והגדרה של כל הפעולות שמשתמש קצה יכול לבצע במערכת תפעולית או במחסן הנתונים. הגדרה זו מתייחסת לכל היישומים שמותר למשתמש להפעיל ולפעולות שמותר לו לבצע (קריאה, עדכון). בסביבת מחסן נתונים מקובל להגדיר למשתמש את הטבלאות שהוא מורשה לגשת אליהן, הטבלאות המדומות שהוא יכול להפעיל, כלי שאילתות שמותר לו להשתמש בהם, משאבי מערכת שמותר לו לצרוך וכד׳.		פרופיל משתמש קצה	End User Profile
מערכת ארגונית המיועדת לתמיכה בתהליכי קבלת החלטות ונוצר כתוצאה מחילוץ נתונים ממערכות תפעוליות שונות של הארגון לאחר טיוב והמרת	53	מחסן נתונים ארגוני	Enterprise Data Warehouse

תיאור המונח	עמוד	מונח עברי	מונח לועזי
הנתונים למבנה ייעודי ועקבי. בסיס נתונים זה מאורגן בדרך כלל לפי נושאים עסקיים שונים ומכיל נתונים בעלי עומק היסטורי כנדרש על ידי מקבלי ההחלטות.			
המרכיב הבסיסי ביותר המנוהל במערכת מידע. דוגמאות ליישויות: עובד, סטודנט, פריט, הזמנה, תנועה בחשבון, קורס באוניברסיטה, ספר בספריה וכד'. לכל יישות יש תכונות (Attributes).	130	יישות	Entity
שיטה להצגת מודל נתונים בצורה ויזואלית תוך שימוש במספר קטן של סימבולים גרפיים. שיטה זו נפוצה מאוד בתיאור מודלים לוגיים של נתונים ומשמשת את מנתחי המערכות ומנהלי בסיסי הנתונים. התרשימים מציגים את אוסף כל היישויות בבסיס הנתונים, את התכונות של כל יישות, את המפתחות ואת הקשרים בין היישויות השונות.	126	תרשים יישויות-קשרים	Entity Relationship Diagram (ERD)
ראה Entity Relationship Diagram.		תרשים יישויות-קשרים	ERD
מערכות מיוחדות המיועדות להצגת מידע למנהלים בכירים. מערכות אלו מכילות בדרך כלל אוסף של סוגי מידע המוכן מראש ונמצא במבנה המתאים למנהלים הבכירים. מערכות אלו מפעילות גרפיקה עסקית, מפות גיאוגרפיות להצגת מידע מרחבי, שיטות מיוחדות להצגת חריגים כמו רמזורים וצבעים. כלים אלה מאפשרים מעבר נוח מרמת סיכום גבוהה לרמות פרטניות יותר בהתאם לדרישות המיוחדות של כל מנהל.	43	מערכות מידע למנהלים	Executive Information Systems (EIS)
לכל נתון בסביבת מחסן נתונים יש תאריך המתייחס למועד שבו הנתון נגזר מהמערכת התפעולית. מידע זה הינו בעל חשיבות רבה למשתמשי מחסן הנתונים. תאריך חילוץ הנתונים מנוהל בקטלוג מחסן הנתונים.	237	תאריך החילוץ	Extract Date
התדירות שבה מתבצעים חילוצי נתונים מהמערכות התפעוליות אל מחסן הנתונים. לסוגי נתונים שונים יש תדירויות גזירה שונות. לדוגמא את נתוני המכירות ניתן לגזור כל שבוע ואילו את נתוני ההזמנות רק פעם בחודש.	237	תדירות החילוץ	Extract Frequency

מונח לועזי	מונח עברי	עמוד	תיאור המונח
Extract Logic	לוגיקת החילוץ	237	הלוגיקה המגדירה כיצד מחלצים נתונים מהמערכות התפעוליות. לוגיקה זו יכולה להיות פשוטה מאוד או מורכבת מאוד, על פי סוג הנתונים ומבנה המערכות התפעוליות.
Fact	עובדה	85 135	הנתונים הבסיסיים המנוהלים במודל הרב-מימדי נקראים עובדות, כי הם מתייחסים לביצועים בפועל של העסק. לדוגמה, כמה מוצרים מסוג מסוים נמכרו ביום מסוים בסניף מסוים, או מה משך השיחה של לקוח מסוים מיעד נתון ליעד אחר בשעה וביום מסוים.
Fact Table	טבלת עובדות	135	הטבלה המרכזית במבנה סכמת הכוכב. היא מכילה את הנתונים המקשרים בין טבלאות המימד ואת הנתונים הנוספים הרלוונטיים.
Foreign Key	מפתח זר	138	עמודה בטבלה כלשהי המשמשת כמפתח עיקרי בטבלה אחרת כלשהי.
Fuzzy Logic	לוגיקה עמומה	78 196	אוסף חדש יחסית של מערכות תוכנה המבוססים על תורת הלוגיקה העמומה, להבדיל מהלוגיקה הרגילה של אמת ושקר. אלגוריתמים אלה מסוגלים לבצע חיפושים מתוחכמים בנתונים ולמצוא מצבים קרובים למצב מסוים, ולא רק מצבים התואמים באופן מלא למצב מסוים.
Gateway	תוכנת שער		תוכנה מיוחדת המאפשרת גישה מסביבה מסוימת אל סביבה מחשובית אחרת. לדוגמה, תוכנה המאפשרת ביצוע פקודות SQL מתוך בסיס נתונים Oracle הפועל על שרת Unix לטיפול בבסיס נתונים DB2 הפועל על מחשב מרכזי. מקובל להשתמש בתוכנות אלו כדי לאפשר נגישות של כלי הגזירה אל בסיסי נתונים וקבצים במערכות מחשוב שונות.
Granularity	רמת פירוט	151 298	תכונה המתארת את פירוט הנתונים המנוהלים בטבלה כלשהי. ככל שרמת הפירוט גבוהה יותר מקובל לומר שה- Granularity גבוהה יותר.
HOLAP	מערכת רב-מימדית מעורבת		ראה Hybrid OLAP.
Hybrid OLAP (HOLAP)	מערכת רב-מימדית מעורבת	109 117	מערכת OLAP המשלבת בסיס נתונים רב-מימדי ייעודי עם בסיס נתונים טבלאי באופן שקוף מבחינת המשתמש.
Hypercube	קוביית על	90 115	מוצר OLAP המנהל את כל הנתונים בקוביה רב-מימדית אחת.

מונח לועזי	מונח עברי	עמוד	תיאור המונח
Inflow Process	תהליכי הזרימה פנימה	231 233	אוסף התהליכים המטפלים בזרימה השוטפת של נתונים מהמערכות התפעוליות או החיצוניות, פנימה אל מחסן הנתונים.
Information	מידע	66	נתונים שעברו עיבוד כלשהו ומשמשים את בני האדם למטרות מסוימות.
Information	מידע	66 78	תוצאה של עיבוד כלשהו של הנתונים הגולמיים המתבצע בהקשר מסוים ומתוך מטרה להגדיל את הידע של המשתמש. המידע הינו הפלט של תהליך עיבוד וניתוח הנתונים.
Information Center	מרכז מידע	44	תפיסה שעסקה בהעברת נתונים ממערכות תפעוליות לסביבה מיוחדת לשאילתות ולניתוח נתונים. תפיסה זו קדמה לתפיסת מחסני הנתונים ופעלה בעיקר בסביבות של מחשבים מרכזיים, ומבוססת על עקרונות דומים, אם כי על טכנולוגיית מחשוב ישנה יותר.
Join	צירוף	74 214	פעולה נפוצה במודל הנתונים הטבלאי, המאפשרת צירוף של שתי טבלאות על בסיס של עמודה אחת או יותר המשותפות לשתיהן.
Knowledge Based Systems	מערכות מבוססות ידע	78	אוסף כלים המבוסס על תהליך הנדסת ידע, שבו אוספים את הידע ממומחה אנושי, מנסחים אותו למערכת המחשוב, ומאפשרים לה על בסיס ידע זה לאתר מצבים שונים בנתונים. מקובל לקרוא למערכות אלו בשם "מערכות מומחה".
LAN	רשת תקשורת מקומית		Local Area Network.
Managed Query Environment	סביבה מנוהלת של שאילתות	75 79	משפחה של כלי שאילתות, המאפשרים בנוסף לביצוע השאילתות, לבצע פעולות ניהול נוספות, כגון שמירת שאילתות, הגדרת הרשאות, תרגום מונחים ממונחים טכניים למונחים עסקיים המוכרים על ידי המשתמשים, הגדרת טבלאות מדומות (Views) לפישוט הגישה לנתונים ועוד.
Market Basket Analysis	ניתוח סל קניות	188	שיטת ניתוח מקובלת ברשתות שיווק שבה מנתחים את תכולת הרכישות של הלקוחות על מנת להבין מהם המוצרים הנרכשים ביחד עם איזה מוצרים.
Massive Parallel Processing (MPP)	עיבוד מקבילי רב עוצמה	209	ארכיטקטורת חומרה מיוחדת המבוססת על מספר רב של מעבדים (מאות ולעיתים גם אלפים) הפועלים במקביל, כאשר לכל אחד מהם זיכרון פרטי. מקובל גם לקרוא לארכיטקטורה זו Shared Nothing. המעבדים מחוברים ביניהם בערוץ תקשורת מהיר ומיוחד.

תיאור המונח	עמוד	מונח עברי	מונח לועזי
ראה Multi Dimensional Data Base.		בסיס נתונים רב מימדי	MDB
מידע על נתונים. בסביבת מחסן הנתונים, המידע המנוהל על הנתונים מגוון מאוד ויכול להתייחס לנושאים שונים, כגון מבנה הטבלאות, עדכניות הנתונים, צורת חילוץ הנתונים, מיפוי מונחים טכניים למונחים עסקיים, כללים לחישוב נתונים תוצאתיים ועוד. מידע זה מנוהל בקטלוג של מחסן הנתונים.	55 256 259	מידע על הנתונים	Meta Data
אוסף מובנה של צעדים, עקרונות, תהליכים ופעולות שיש לבצע כדי לממש נושא כלשהו. ראה גם Data Warehouse Methodology.	283	מתודולוגיה	Methodology
כינוי למשפחה של מוצרי תוכנה המשמשים כתווך בין מרכיבי תוכנה שונים ומבוזרים. תוכנה זו מאפשרת את הקישוריות ואת הפעולה ההדדית של מרכיבי התוכנה לביצוע המשימות.	31 61 302	תוכנת תווך	Middleware
ראה Multi Dimensional OLAP.		מערכת רב-מימדית מבוססת בסיס נתונים ייעודי	MOLAP
ראה Massive Parallel Processing.		עיבוד מקבילי רב-עוצמה	MPP
ראה Managed Query Environment.		סביבה מנוהלת של שאילתות	MQE
בסיס נתונים מיוחד וייעודי המותאם לניהול מבני נתונים רב-מימדיים. בסיסי נתונים אלה עומדים בבסיסם של מוצרי MOLAP והם אלה אשר מספקים למערכות אלו את התחכום והביצועים שלהם.	107 276	בסיס נתונים רב-מימדי	Multi Dimensional Data Base (MDB)
מערכת תוכנה ייעודית המאפשרת ניהול והצגה רב-מימדית של המידע תוך שימוש בבסיס נתונים רב-מימדי ייעודי (MDDB). מערכות אלו משתמשות במבני נתונים קנייניים לניהול הנתונים ולכן מגיעות לזמני תגובה טובים מאוד. החיסרון העיקרי של שיטה זו בצורך לטעון את הנתונים מראש, והמגבלות לגבי נפח הנתונים שניתן לנהל בהן. ראה גם OLAP.	109 110	מערכת רב-מימדית מבוססת בסיס נתונים ייעודי	Multi Dimensional OLAP (MOLAP)

תיאור המונח	עמוד	מונח עברי	מונח לועזי
מחסן נתונים המבוסס על מספר שכבות: שכבת מערכות תפעוליות, שכבת מחסן נתונים ארגוני ושכבת מרכולי נתונים ייעודיים. ארכיטקטורה זו היא המומלצת ביותר, אולם קשה למימוש.	57	מחסן נתונים רב-שכבתי	Multi Tier Data Warehouse
תהליך המבוצע על ידי המשתמש, ובאמצעותו הוא עובר בין הנתונים השונים תוך החלפת מימדים, רמת פירוט וכד׳.	127	ניווט	Navigation
שיטה להצגה של מספר מימדים רב בטבלה דו-מימדית רגילה. הכותרת האופקית או האנכית מכילה היררכיה של מימדים שונים.	95	מימדים מקוננים	Nested Dimensions
מערכות תוכנה המשמשות לכריית נתונים ומבוססות על מודל רשת עצבית, המהווה חיקוי מסוים ומוגבל של צורת הפעולה של מוח אנושי. מערכות אלו טובות מאוד לפילוח אוכלוסיות ומתאימות למצבי חיזוי, כאשר הנתונים אינם נקיים לחלוטין.	77 193	רשתות עצביות	Neural Networks
תכונות המנוהלות בטבלת העובדות, אולם אין כל משמעות לביצוע פעולות אריתמטיות עליהן.	136	תכונות שאינן ניתנות לחיבור	Non Additive Attributes
ארכיטקטורת חומרה חדשה יחסית המבוססת על מספר רב של מעבדים הפועלים במקביל, שלכל אחד מהם זיכרון פרטי וגישה לזיכרון משותף. ארכיטקטורה זו מנסה לשלב את הטוב שבשתי הארכיטקטורות SMP ו-MPP.	211	ארכיטקטורת זיכרון לא אחידה	Non Uniform Memory Architecture (NUMA)
כינוי למצב שבו הנתונים אינם מתעדכנים כתוצאה מארועים שוטפים. מקובל לומר שבסיס הנתונים של מחסן הנתונים הוא Nonvolatile מאחר ואינו משתנה, אלא כתוצאה מטעינה חדשה של נתונים.	51	לא מתעדכן	Nonvolatile
ראה Non Uniform Memory Architecture.		ארכיטקטורת זיכרון לא אחידה	NUMA
ראה Open Data Base Connectivity.		קישוריות פתוחה בין בסיסי נתונים	ODBC
ראה Operational Data Store.		מאגר נתונים תפעולי	ODS
ראה On Line Analytical Processing.		מערכת רב-מימדי	OLAP

תיאור המונח	עמוד	מונח עברי	מונח לועזי
ראה On Line Transaction Processing.		עיבוד תנועות מקוון	OLTP
מערכת תוכנה ייעודית המאפשרת ניהול והצגה רב-מימדית של המידע, תוך שימוש בצורת הצגה מיוחדת ובמספר אופרטורים מיוחדים.	75 105	מערכת רב-מימדית	On Line Analytical Processing
מערכות תוכנה המאפשרות ביצוע של תנועות עיבוד בצורה מקוונת. שם כולל למערכות התפעוליות של הארגון התומכות בביצוע מיגוון התנועות העסקיות המבוצעות באופן שוטף.	34 125	עיבוד תנועות מקוון	On Line Transaction Processing
תקן דה-פקטו שפותח ומופץ על ידי חברת מיקרוסופט ומגדיר אוסף ממשקי תכנות שיאפשרו גישה מסביבת תוכנה כלשהי לבסיסי נתונים שונים, מבלי שהיישום יכיר כיצד כל אחד מהם פועל למעשה. תקן זה הוא נגזרת מסוימת של תקן שפותח על ידי הארגון SQL Access Group.	74	קישוריות פתוחה בין בסיסי נתונים	Open Data Base Connectivity (ODBC)
מאגר נתונים הדומה באפיונים שלו למחסן נתונים, אולם מכיל נתונים הנדרשים על ידי המערכות התפעוליות, ולכן רמת העדכניות שלו גבוהה מאוד והעומק ההיסטורי שלו קטן יחסית. ניתן לראות בארכיטקטורה זו תוספת למערכות תפעוליות, על ידי הוספת שכבה שמאפשרת אחזור נוח של נתונים תפעוליים באמצעות כלי קצה מתוחכמים.	59	מאגר נתונים תפעולי	Operational Data Store (ODS)
מערכות מידע המשמשות את הארגון בתפעול השוטף.	34 69	מערכות תפעוליות	Operational Systems
אוסף תהליכים המטפלים בזרימת הנתונים ממחסן הנתונים אל המשתמשים ואל כלי ניתוח והצגת הנתונים.	256	תהליכי הזרימה החוצה	Out Flow Process
עמודה אחת או יותר המזהות באופן חד ערכי שורה בטבלה.	138 139	מפתח עיקרי	Primary Key
כלי לבניה מהירה ונוחה של שאילתות ע״י משתמשים.	41 74	מחולל שאילתות	Query Generator
ראה Relational Data Base Management System.		מערכת לניהול בסיס נתונים יחסי	RDBMS
מצב בו נתון אחד, המייצג עובדה כלשהי, מאוחסן מספר פעמים בתוך בסיס הנתונים. לדוגמה שמירת כתובת הלקוח במספר טבלאות שונות.		כפילות נתונים	Redundant Data

מונח לועזי	מונח עברי	עמוד	תיאור המונח
Referential Integrity	אמינות הקשרים בין טבלאות		אוסף אילוצים המגדירים כיצד לשמור על האמינות והשלמות של קשרים בין טבלאות שונות במודל הטבלאי. מגדיר קשר של אחד לרבים בין שתי טבלאות.
Relational Data Base Management System (RDBMS)	מערכת לניהול בסיס נתונים טבלאי	107 112 125 276	מערכת תוכנה המאפשרת ניהול של בסיס נתונים המבוסס על מודל הנתונים הטבלאי. תוכנה זו מנהלת את הנתונים באוסף הטבלאות ומספקת מיגוון אופרטורים המבוססים על שפת SQL לביצוע כל הפעולות הנדרשות על הנתונים.
Relational OLAP (ROLAP)	מערכת רב-מימדית מבוססת בסיס נתונים טבלאי	109 113	מערכת תוכנה ייעודית המאפשרת ניהול והצגה רב-מימדית של המידע תוך שימוש בבסיס נתונים טבלאי רגיל המעוצב במבנה מיוחד - עיצוב רב-מימדי. קיימות ארכיטקטורות שונות למימוש המבט הרב-מימדי המבוסס על בסיס נתונים טבלאי. היתרון במערכות אלו הוא העדר הצורך לטעון מבנה נתונים ייעודי והיכולת לפעול על בסיסי נתונים ענקיים. חיסרון השיטה בביצועים פחות טובים ובתחכום קטן יותר של האופרטורים והפעולות שניתן לבצע. ראה גם OLAP.
Replication	שכפול	58	שיטה לביזור בסיסי נתונים על ידי שכפול טבלאות של בסיס נתונים למספר שרתים שונים, כדי לאפשר גישה מקומית לנתונים שיכולים להתעדכן בשרתים מרוחקים. השיטה מתבססת על הפצת השינויים בלבד, כדי לייעל את ההפצה ולהקטין ככל האפשר את העומס על רשת התקשורת.
Report Generator	מחולל דוחות	41 74	רכיב תוכנה המיועד לפיתוח מהיר וידידותי של דוחות מתוך בסיס הנתונים. המחולל מאפשר הגדרה מהירה של הטבלאות והעמודות הנדרשות, את התנאים לאחזור הנתונים, את סדר המיון ובמידת הצורך - גם את הלוגיקה הנוספת לחליצת הנתונים ועיבודם. רכיבי תוכנה אלה נקראים לפעמים Report Writer.
Repository	מילון נתונים	256 259 302	בסיס נתונים המכיל נתונים אודות הנתונים המנוהלים בבסיס הנתונים, או כמו שמקובל לומר: מידע על הנתונים. מילון הנתונים מכיל את כל המידע הדרוש להגדרת הנתונים: שם הנתון, סוג הנתון, צורת הייצוג שלו, מידע על השימוש בנתונים, כללים ליצירת הנתון, מידע על המורשים לראות או לעדכן את הנתון וכד'.

מונח לועזי	מונח עברי	עמוד	תיאור המונח
ROLAP	מערכת רב-מימדית מבוססת בסיס נתונים טבלאי		ראה Relational OLAP.
Roll Down			ראה Drill Down.
Roll Up			ראה Drill Up.
Scalability	כושר גידול	219 229	תכונה של מערכת לתמוך בהיקפי נתונים ובעיבודים ההולכים וגדלים, ללא פגיעה מהותית בביצועיה ותוך שמירה על יחס סביר של עלות לביצועים.
Schema	סכמה		אוסף לוגי של אובייקטים (טבלאות, עמודות, אינדקסים וכד׳) בבסיס נתונים המאורגנים יחד. לכל סכמה יש מורשה שיכול להעניק הרשאות למשתמשים נוספים לגשת לאובייקטים של בסיס הנתונים.
Semi Additive Attributes	תכונות בנות חיבור למחצה	136 173	תכונות נומריות המנוהלות בטבלת העובדות, שניתן לבצע עליהן פעולות אריתמטיות מסוימות בלבד.
Slicing and Dicing	פריסה וחיתוך	90	פעולה נפוצה בכלי OLAP המאפשרת להתבונן על מבנה הנתונים הרב-מימדי מנקודות מבט שונות, כלומר פעולה המאפשרת את החלפת המימדים המוצגים, על פי הצורך.
SMP	עיבוד מקבילי סימטרי		ראה Symmetrical Multi Processing.
Snowflake Schema	סכמת פתיתי שלג	155	הרחבה מסוימת של סכמת כוכב, המאפשרת ניהול של טבלאות מימד קטנות סביב טבלאות המימד העיקריות. ראה Star Schema.
Sparcity	רמת דלילות	102 177	מונח המתאר את העובדה שלא כל התאים בקוביה רב-מימדית מכילים נתונים. לצורך ייעול ניהול מבנה הנתונים, המערכות הרב-מימדיות מכירות בעובדה שהקוביות בדרך כלל דלילות ולכן חוסכות במקום ובניהול של תא ריק.
SQL (Structured Query Language)	שפת גישה לבסיסי נתונים טבלאיים	74	שפת הגישה המקובלת לבסיסי נתונים טבלאיים. קיים תקן רשמי לשפה המפורסם מעת לעת על ידי מכון התקנים האמריקאי (ANSI). ראה גם Relational Data Base Management System.
SQL Query Tool	כלי שאילתות מבוסס SQL		כלי שאילתות התומך בגישה לבסיסי נתונים טבלאיים. כלים אלה מבוססים בדרך כלל על

מונח לועזי	מונח עברי	עמוד	תיאור המונח
			ממשק גרפי קל וידידותי, המאפשר למשתמש לנסח את השאילתה ללא ידיעת התחביר של שפת SQL. כלים אלה מפתחים באופן אוטומטי את משפט SQL האופטימלי לביצוע הבקשה.
Star Query	שאילתת כוכב	141	שאילתה המתבצעת על סכמה במבנה כוכב. רכיב האופטימיזציה של SQL מכיר את מבנה הכוכב, ולכן מבטיח חילוץ שורות מתוך טבלאות המימד לפני שמבצע את פעולת הצירוף (Join) עם טבלת העובדות, שהיא בדרך כלל הטבלה הגדולה ביותר.
Star Schema	סכמת כוכב	139	מבנה מיוחד של סכמת בסיס נתונים טבלאי האופייני לסביבות מחסני נתונים ומתאים לניהול נתונים רב-מימדיים בצורה נוחה ויעילה. הנתונים העיקריים מנוהלים במרכז הכוכב הכוללת את טבלת העובדות וסביבה אוסף של טבלאות מימדים.
Subject Oriented	מונחה נושא	47 50	שיטה לארגון וניהול נתונים בהתאם לנושא עסקי - לקוחות, מוצרים, הזמנות וכד׳ - ולא על פי התהליכים התפעוליים.
Subject Oriented Data Base	בסיס נתונים נושאי	50	בסיס נתונים המאורגן לפי נושא, ולא לפי יישום. לדוגמה: בסיס נתונים שיווקי, בסיס נתונים לתקלות וכד׳. מחסן הנתונים הארגוני מורכב מאוסף של בסיסי נתונים המאורגנים לפי נושאים.
Summary Table	טבלה סיכומית		ראה Aggregate Table.
Surrogate Key	מפתח מלאכותי	142 143 251	עמודות מפתח שמקובל להוסיף אותן לטבלאות בסכמת כוכב, כדי ליצור קשרים בין טבלאות המימדים לבין טבלת העובדות. מפתחות אלה מיוחדים לסביבת מחסן הנתונים לעומת המפתחות הטבעיים של הטבלאות האופייניים לסביבות תפעוליות.
Symmetrical Multi Processing (SMP)	עיבוד מקבילי סימטרי	206	ארכיטקטורת חומרה מיוחדת המבוססת על מספר רב של מעבדים הפועלים במקביל, אך מנצלים את אותו זיכרון מרכזי. מקובל גם לקרוא לארכיטקטורה זו Shared Everything. זוהי ארכיטקטורה נפוצה מאוד ונתמכת על ידי מספר רב מאוד של יצרנים.
Target Data Base	בסיס נתונים יעד		בסיס הנתונים המהווה את היעד של כלי חילוץ הנתונים ושל כלי שכפול הנתונים. בדרך כלל זהו שם נוסף לבסיס הנתונים של מחסן הנתונים.

תיאור המונח	עמוד	מונח עברי	מונח לועזי
מחסן נתונים המורכב משלוש שכבות: שכבת המערכות התפעוליות, שכבת מחסן הנתונים הארגוני ושכבת מרכולי הנתונים. מקובל לקרוא לארכיטקטורה זו גם Multi Tier Data Warehouse.	58	מחסן נתונים תלת שכבתי	Three Tier Data Warehouse
אחד מהמימדים הבסיסיים והחשובים ביותר בכל מחסן נתונים. מנהל באופן ישיר את ציר הזמן ומאפשר לבצע השוואות רגישות לזמן.	146	מימד הזמן	Time Dimension
מצב בו הנתונים רגישים לציר הזמן, כלומר צריכה להיות אפשרות להציג את נכונות הנתונים על ציר הזמן.	51	רגיש לזמן	Time Variant
ראה Transaction Processing Council.		המועצה לעיבוד תנועות	TPC
יחידת עבודה בסיסית המייצגת פעולה בסיסית מבחינה עסקית. כל פעולה מעדכנת בדרך כלל מספר טבלאות ולכן סיום לא תקין שלה יכול לשבש את אמינות בסיס הנתונים. כל המערכות לניהול בסיס הנתונים מכילים מנגנונים מיוחדים המבטיחים את סיומה המוצלח של התנועה, ובמקרה של תקלה הן מאפשרות את מחיקת כל השינויים שכבר נעשו. המערכות התפעוליות מעדכנות את בסיסי הנתונים על ידי ביצוע אוסף של תנועות. דוגמאות לתנועות: העברה של סכום מחשבון לחשבון, כניסה של פריט למלאי, משלוח פריט ללקוח.	34 65 84 126	תנועה	Transaction
ארגון וולונטרי ובלתי תלוי העוסק בהגדרת מבחני ביצועים של מערכות לניהול בסיסי נתונים. ארגון זה בנה מבחני ביצועים שונים והידועים ביניהם: TPC-C עבור מערכות הפועלות בסביבות עתירות תנועות, ו- TPC-D עבור מערכות הפועלות בסביבות מחסני נתונים.	228	המועצה לעיבוד תנועות	Transaction Processing Council
מחסן נתונים המורכב משתי שכבות, שכבת המערכות התפעוליות ושכבת מחסן הנתונים הארגוני.	55	מחסן נתונים דו-שכבתי	Two Tier Data Warehouse
אוסף התהליכים המטפלים בבניית סיכומים (Aggregates) שונים הנדרשים לקבלת זמני תגובה משופרים לשאילתות.	231 254	תהליכי הזרימה מעלה	Upflow Process
אוסף אופרטורים מתמטיים הפועלים על וקטור שלם, להבדיל מהאופרטורים הפועלים על מספרים בודדים (Scalars).	92	אריטמטיקה וקטורית	Vector Arithmetics

מונח לועזי	מונח עברי	עמוד	תיאור המונח
Verification Based Analysis	ניתוח מבוסס אימות	72 181	תהליך ניתוח נתונים המבוסס על אימות הנחות שהמשתמש מניח ומנסה לאמת תוך כדי הניתוח. רוב כלי השאילתות הרגילות וגם כלי ה OLAP מבוססים על מודל ניתוח נתונים זה.
Very Large Data Bases (VLDB)	בסיסי נתונים גדולים	127 299	ענף של מערכות ניהול בסיסי הנתונים העוסק באתגרים המיוחדים של ניהול בסיסי נתונים ענקיים. ענף זה מטפל באופטימיזציה של שאילתות לשיפור הביצועים, בשיטות לחלוקת העיבוד של בסיס הנתונים בין מעבדים שונים, בטכניקות לחלוקת בסיס הנתונים למחיצות, באתגרים מיוחדים בגיבוי ושחזור בסיסי נתונים ענקיים וכדי.
View	נקודת מבט	74	אפשרות לספק למשתמש נקודת מבט על הנתונים. מקובל לקרוא לנקודת המבט גם טבלה מדומה.
Virtual Data Warehouse	מחסן נתונים מדומה	53 61	שיטה המנסה לדמות מערכת מחסן נתונים ע"י מתן אפשרות לגישה ישירה לנתונים של המערכות התפעוליות באמצעות אוסף של כלי שאילתות.
Visualization Techniques	טכניקות ויזואליזציה	77 187	אוסף של כלים וטכניקות המאפשרים הצגה מתוחכמת של נתונים. לדוגמה: הצגה של פיזור נתונים, הצגה תלת-מימדית, שילוב של מידע גיאוגרפי עם מידע נומרי, אנימציה של הצגת הנתונים על ציר הזמן וכדי.
VLDB	בסיסי נתונים גדולים		ראה Very Large Data Bases.
WAN	רשת תקשורת מרחבית		Wide Area Network.
Warehouse Keys	מפתחות מחסן נתונים		ראה Surrogate Keys.

נספח ב׳: ביבליוגרפיה

Anahory Sam, Murray Dennis - Data Warehousing In The Real World, Addison-Wesley, 1997

Bokun M., Taglienti C. - Incremental Data Warehouse Updates: Approaches and Startegies for capturing changed data, DM Review Magazine, May 1998

Brand E, Gerritsen R. - Decision Trees, DBMS, Data Mining Solutions Supplement, July 98

Brobst S., Robertson O. - Taming Data Giants: Getting a Grip on VLDB in the Real Wold, DBMS, February 1997

Brooks Peter - Data Mining Today, DBMS, February 1997

Burleson C.L., Tabler D. E. - Data Warehousing: Putting it All Together, The Data Warehousing Institute, 1995

Business Objects - White Paper: Query Tools and Your Data Warehouse

Darling Charles B. - How To Integrate Your Data Warehouse, Datamation, May 1996

Debevoise Nielson - The Data Warehouse Method, Prentice Hall, 1998

Devlin B., Cote L.D. - Data Warehouse: From Architecture to Implementation, Addison Wesley, 1996

Dyche Jill - Scoping Your Data Mart Implementation, DBMS, August 1998

Edelstein Herb - Data Mining: Exploiting the Hidden Trends in Your Data, DB2 Magazine, Spring 1996

Hackathorn Richard - Criteria for Successful Data Warehousing, The Data Warehousing Institute, 1995

Hackathorn Richard - Data Warehouse’s Credibility Crisis, Byte, August 1997

Hackney Douglas - Data Warehouse Delivery, Data Management Review, February 1998

Hurwitz Group - White Paper: Scalable Enterprise OLAP Requirements - Data Depths and Breadth

Hyperion Software - White Paper: The Role of the OLAP Server in a Data Warehousing Solution

Inmon W.H. - Does Your Datamart Vendor Care About Your Architecture, Datamation, March 1997

Inmon W.H. - What is a Data Mart, D2K Inc Publishing, 1996

Inmon W.H., Hackathorn R.D. - Using the Data Warehouse, John Wiley & Sons, 1994

Inmon W.H., Kelley C. - Rdb/VMS: Developing the Data Warehouse, QED Publishing Group, 1993

Inmon W.H., Rudin K., Buss C. - Data Warehouse Performance, John Wiley & Sons, 1998

Kimball Ralph - A Dimensional Modeling Manifesto, DBMS, August 1997

Kimball Ralph - The Data Warehouse Toolkit, John Wiley & Sons, 1997

Kimball Ralph - Data Warehouse Architect series of articles in DBMS magazine

Kimpball R., Reeves L. - The Data Warehouse Lifecycle: Expert Methods for Designing, Developing and Deploying Data Warehouses, John Wiley & Sons, 1998

Mattison Rob - Data Warehousing: Strategies, Technologies and Techiniques, McGraw-Hill, 1998

Meredith M. E., Kahder A. - Divide and Aggregate: Designing Large Warehouses, Database Programming & Design, August 1997

MicroStrategy - White Paper: Relational OLAP - An Enterprise Wide Data Delivery Architecture

Moriarty Terry - A Metadata Management How-To, Database Programing & Design, February 1997

Orr Ken - Data Warehouse Technology, A White Paper, The Ken Orr Institute, 1997

Parsaye Kamran - OLAP & Data Mining: Bridging the Gap, Database Programming & Design, February 1997

Pieter Adriaans, Dolf Zantinge - Data Mining, Addison-Wesley, 1996

Pilot Software - White Paper: An Introduction to Data Mining

Pilot Software - White Paper: An Introduction to OLAP: Multidimensional Terminology and Technology

Platinum Technolgy - White Paper: Mnaging the Data Warehouse Throughout Its Lifecycle

Poe V., Brobst S. - Building a Data Warehouse for Decision Support, Prentice Hall, 1997

Raden Neil - Database Design for Data Warehousing, The Data Warehousing Institute,1995

Rudin Ken - Where Parallel Lines Meet, Byte, May 1998

Silverston L., Graziano K. - The Data Model Resource Book: A Library of Logical and Data Warehouse Models, John Wiley & Sons, 1997

Speedware Corporation - White Paper: Hybrid OLAP - The Best of Both Worlds

Ullman J.D. - Principles of Database and Knowledge-base Systems, Volume I, Computer Science Press, 1988

Ullman J.D. - Principles of Database and Knowledge-base Systems, Volume II, Computer Science Press, 1989

Watterson Karen - Parallel Tracks, Datamation, May 1997